SIXTH FRAMEWORK PROGRAMME
PRIORITY [#]
[Information Societies Technology]

Contract for:

SPECIFIC TARGETED RESEARCH PROJECT

Annex I - "Description of Work"

Project acronym: *RODIN*
Project full title: *Rigorous Open Development Environment for Complex Systems*
Proposal/Contract no.: 511599
Related to other Contract no.: *(to be completed by Commission)*

Date of preparation of Annex I: *April 27, 2004*

Operative commencement date of contract: *(to be completed by Commission)*

TABLE of CONTENTS

1. Project summary

Our overall objective is the creation of a methodology and supporting open tool platform for the cost effective rigorous development of dependable complex software systems and services.

We focus on tackling complexity

- caused by the environment in which the software is to operate
- which comes from poorly conceived architectural structure.

Mastering complexity requires design techniques that support clear thinking and rigorous validation and verification. Formal design methods (FM) do so. Coping with complexity also requires architectures that are tolerant of faults and unpredictable changes in environment. This is addressed by fault tolerance (FT) design techniques.

We will develop a **unified methodology** combining FM with

FT design principles by using a systems of systems approach, where both software and environment are modelled together.

We will tackle complex architectures: our systems approach will support the construction of appropriate abstractions and provide techniques for their structured refinement and decomposition.

We will ensure cost effectiveness, the methods and platform will support reuse of existing software. We will thus extend existing FM with generic mechanisms to support **component reuse and composition**.

Tool support for construction, manipulation and analysis of models is crucial and we will concentrate on a comprehensive **tool platform** which is **openly available** and **openly extendable** and has the potential to set a European standard for industrial FM tools.

The methods and platform will be validated and assessed through industrial case studies.

The novel aspects of this proposal are the pursuit of a systems approach, the combination of FM with FT techniques, the development of FM support for component reuse and composition and the provision of an open and extensible tools platform for formal development. In particular, we believe that the open tools platform will have a significant impact on future research in FM tools and will encourage greater industrial uptake.

2. Project objectives

The overall objective of the RODIN project is the creation of a methodology and supporting open tool platform for the cost effective rigorous development of dependable complex software systems and services.
The adjective “complex” is used throughout this document to reflect the fact that modern computer systems are among the most complicated things designed by humans (as distinct from those things which have evolved over enormous periods of time). Some of the facets of even simple software systems which make them difficult to comprehend include:

- It is non-trivial to design even fairly small programs because they are sequences of instructions to an unquestioning automaton which will execute instructions however silly; it is thus necessary to foresee all possible exceptions.
- It is impossible to fully test even modestly sized programs because the number of paths grows exponentially with the length of the program.
- Any system will need to evolve: there is (in addition to near perfect behaviour at installation) a requirement on a designer to plan in flexibility for a system to change as the requirements evolve.

Emerging systems are becoming ever more pervasive and RODIN is focusing on tackling two forms of extra complexity which arise in these software systems:

- The complexity caused by the environment (hardware, human users, physical processes in the controlled environment, communication links etc.) in which software systems operate: such environments are often prone to faults and unpredictable changes which lead to system failure if appropriate fault tolerance techniques are not deployed.
- The complexity resulting from architectural structure in large software systems: this arises because of the inherent complexity in the services provided by a system and because of poorly-constructed software architecture. Architectural complexity impedes the ability to understand and analyse system behaviour thus making large systems prone to failure and difficult to maintain or evolve. Furthermore, complex architecture can make it almost impossible for a user to safely understand a system.

Coping with complexity requires design techniques that support clear thinking, clean architectures and rigorous validation and verification. Formal design methods, based on sound mathematical principles, meet these requirements. To be acceptable in industry, formal methods must be supported by good tools and must be integrated into existing industrial development processes. Some formal methods, such as VDM, Z and the B Method, have reached high levels of maturity and are already used in industrial settings, though mainly just for the detailed design of relatively small systems or system components, rather than to aid the overall design of large complex systems. The RODIN team includes the originators of these methods who will seek to derive maximum benefit from their experience and offer realistic, usable, rigorous methods.

To address architectural complexity some key techniques and approaches provided by existing formal methods are being used, namely, *abstraction*, *refinement* and *decomposition*. Abstraction is a means for making precise the global properties of a system in the form of models that abstract away from architectural structure and detail.

Refinement supports the addition of functional and architectural detail in a stepwise manner while preserving properties of the more abstract models. After an abstract model has been refined to a sufficient degree, it is decomposed into smaller subsystems which can themselves be further developed separately.

Abstraction, refinement and decomposition together make architectural complexity manageable by allowing structure to be introduced and analysed in a controlled and modular manner. Furthermore, they help to reduce architectural complexity by encouraging cleaner, better thought out designs in which the effects of modifications are more manageable and predictable because new interactions are additive rather than multiplicative.

To address the complexity caused by the environment, the proposed methodology and platform has adopted a *systems approach* to software development. In a systems approach the required control behaviour of a physical system is modelled at a high level of abstraction and the software required to achieve the desired control behaviour is gradually derived and extracted through refinement and decomposition. Abstraction is essential to achieve this since it is infeasible to have complete models of environments. A systems approach also entails the adoption of formal methods early in the development process. The experience of the industrial partners ClearSy and Praxis is that adoption of formal methods in the early stages of a project is much more effective than late or post-implementation usage of formal methods.

RODIN is focusing on a development platform for the construction of fault tolerant systems, safety critical and safety related systems and communications systems. Such systems are required to have a high degree of dependability and typically exhibit both architectural and environmental complexity. Together, the case studies that are being used to guide and validate the research in RODIN cover fault tolerance, safety criticality and communications systems issues. The carefully-selected set of case studies are protocol engineering, engine fault management, mobile telecoms services, air traffic control display system and an ambient campus. The case studies provide material (requirements, specifications, designs) that guides the research in the early phases of the project. In the later phases, the case studies will be used to validate the methods and support platform.

Long term cost effectiveness in systems development is achieved through greater support for reuse of existing developments in new systems. Long term cost effectiveness of the development environment will be achieved through platform openness which will allow other parties to contribute additional tools in the future. To facilitate the integration of the RODIN methods and tools with engineering practice, we are developing a tool-supported bridge between UML and mathematically based notations such as B.

Taking account of the above considerations, the overall objective of RODIN leads to the following specific measurable objectives:

1. A collection of reusable development templates (models, architectures, proofs, components, etc.) produced by the case studies. The goals of the cases studies will be defined in detail by month 6. Initial and intermediate results will be available by months 12 and 24, while a final set of development templates will be available by month 36 of the project.
2. A set of guidelines on a systems approach to the rigorous development of complex systems, including design abstractions for fault tolerance and guidelines on model

mapping, architectural design and model decomposition. Initial and intermediate guidelines will be available by months 12 and 24, with the final versions by month 36.

3. An open tool kernel supporting extensibility of the underlying formalism and integration of tool plug-ins. Open specification of the kernel will be made publicly available by month 12 of the project. Prototypes of the basic tools will be available by month 18. Full working versions will be available by month 30, with final versions being ready by month 36.
4. A collection of plug-in tools for model construction, model simulation, model checking, verification, testing and code generation. Open specification of the plug-in tools will be made publicly available by month 12 of the project. Prototype versions of the plug-in tools will be available by month 18, while final versions will be available by month 36.

The research and development required to reach these objectives is directed by the following criteria:

a). The methods and platform should support a systems approach to software development.

A systems approach must provide guidelines on design abstractions for modelling and analysis of the software under development together with its environment. In order to map high-level designs down to realisable architectures, the systems approach must provide sound techniques allowing high-level models to be decomposed into separate communicating subsystems on which further formal analysis and architectural decomposition may be performed separately. The output of a system design process might be a formally proved architecture, that is, a system decomposition that is correct with respect to a high level system specification, or a formally proved implementation. For pragmatic reasons, the techniques we develop build on the B Method as it facilitates a systems approach to development and has a high level of maturity in terms of tool support and industrial usage. We also support development in a combination of UML and B.

b). The methods and platform should support the development of systems that are tolerant of faults and unpredictable changes in the environment.

A common cause of the poor quality of existing systems is the disregard of fault tolerance techniques in the development process. Often fault tolerance is introduced too late and in an unsystematic way. Such an ac-hoc approach is not able to guarantee that the system can deal properly with the wide variety of faults which the system environment can cause or be susceptible to. It leads to significant impairment of a system's dependability – the degree of reliance which we can justifiably place on the system. We address this by developing a unified fault tolerant and formal design methodology, comprising amongst other aspects a set of guidelines and reusable formal design templates for fault abstractions and fault-tolerance refinement supported by the specialized features of the development platform.

c). The methods and platform should support reuse of existing software developments.

Components that are universally reusable across domains would be difficult to achieve. Instead we focus on reusing components within domains that can be tailored and

combined to construct a system for a specific market or customer. Examples include components that provide the typical functionality required for engine control systems or a mobile communications network. Identifying when components can be reused is difficult and we see abstract formal models as an essential way of facilitating this. Our thesis is that an abstract model of a software component makes it easier to identify a component as being usable for a specific development and we are testing this thesis in the project. To be most effective, we need to support reuse of any formal models and formal analysis associated with software components. An important challenge is the development of language constructs and tools that support such reuse.

d). The tools supporting the methods should be integrated on a single platform and that platform should be open to extension by other parties.

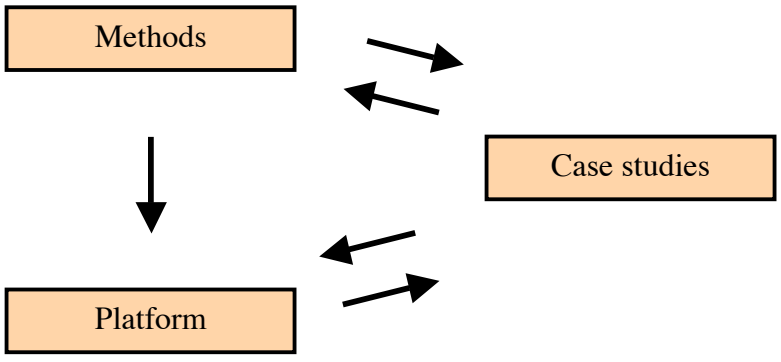
Tool support is essential for any methodology, formal or not, to be applied or even tested in industry. One of the key points of RODIN is thus the development of a single tool platform, supporting our unified methodology. To maximise the impact of our research, the platform should be openly available and extensible both by third parties and individual partners of the project. We thus provide an open development platform with well defined intermediate formats to allow new tools to be added as plug-ins. To this end we are developing an intermediate (XML-based) representation and use a generic open source development platform (such as Eclipse) as the basis on which to build. We are building on the partners' extensive expertise in building earlier tools for B, VDM and other notations. We also provide a bridge between UML tools and the RODIN platform.

e). The methods and platform should be properly validated and assessed through industrial case studies.

The case studies being used in RODIN are provided by the partners who are key industrial players and potential end-users of the RODIN technology. They are realistic industrial case studies that are guiding and strongly testing the RODIN technology.

These outcomes will be of value to the industrial partners and to other similar players and will also make an impact on the systems engineering research community.

The interplay between methods, tools and case studies is at the heart of the RODIN plan. The arrows on the following diagram show the influences which we will strive to utilise within the project in order to achieve outcomes of the project which have maximum applicability.



The project is structured into the following seven workpackages:

WP1. Research drivers
WP2. Methodology
WP3. Open tool kernel
WP4. Modelling and verification plug-ins
WP5. Dissemination and exploitation
WP6. Project management
WP7. Project review and assessment

The project builds on experience gained in developing industrial case studies and recent experience in industrial applications of formal methods. It also builds on further advances in theoretical research underlying the methodology to be developed and experience of developing formal methods tools. The research is guided by case studies provided by the industrial partners and by the experience industrial and academic partners have accumulated in applying formal methods and existing tools. This work is being undertaken by a consortium with a careful balance of academic and industrial partners. ClearSy and Praxis have considerable experience with the industrial application of formal methods. VTEC is less experienced in using formal methods but is looking for serious involvement in their use with a view to adoption. Nokia has substantial experience in applying formal methods but is keen to make application of current technologies (such as UML and MDA) more formal. Several partners have experience of developing research tools for formal methods while ClearSy have successfully commercialised the Atelier B tool.

The consortium includes partners from relevant Framework 5 projects including DSoS (Dependable Systems of Systems), MATISSE (Methods And Tools for Industrial Strength Systems Engineering) and PUSSEE (Paradigm Unifying System Specification Environments for proven Electronic design) and RODIN builds on results of these and other relevant projects. DSoS developed structured fault tolerance approaches and produced formal models of fault tolerance and basic dependability concepts. In MATISSE, the B method was successfully used to develop complex systems such as an automated railway braking system, an embedded smart card byte code verifier and an industrial drug-discovery robot. In the PUSSEE project, formal proof of hardware system properties is being introduced through a modular system design methodology that integrates sub-systems co-verification with system refinement and reusability of virtual system components. This is being achieved by combining the UML and B languages to allow the verification of system specifications through the composition of proven sub-systems.

Previous research effort has focused on the use of formal methods at later design stages only and there are no significant results on properly integrating the use of formal methods and fault tolerance and other adaptive mechanisms. RODIN will produce significant new results that were not achieved by the above projects or similar projects. In the area of tools, RODIN will produce a tools platform that is open to extension by us and by other parties in contrast to the monolithic inflexible tools produced by previous research. We will produce plug-ins for the platform that exploit the openness of our platform. RODIN will provide support for rigorous design at a systems level using formal methods and support for the treatment of fault tolerance, of mobility and of flexible architectures as part of the rigorous design process.

Contribution to state of the art

A lot of the development of formal methods over the last 30 years has been led by European researchers leading to various methods including model oriented approaches such as B, VDM, Z, and process algebras such as CSP, CCS, LOTOS and the π-calculus. There have been many successful applications of formal methods in European and international industry most notably in the areas of transportation and hardware design. These successes provide strong evidence that the use of formal methods leads to more dependable systems and, significantly, that their use is cost-effective. However, the industrial application of formal methods has focused on the development of software or hardware components, rather than complete complex systems**.** RODIN's proposal to use a systems approach in the use of formal methods will thus significantly extend the state of the art in formal methods.

Dependability research is also very strong in Europe, and mechanisms and approaches developed by the dependability community have been applied in industry, especially in the safety critical domain. However, the application of formal methods to the rigorous development and analysis of fault tolerance mechanism is not as mature. This is a further area where RODIN will extend the state of the art.

Many analysis tools, such as model checkers and theorem provers, have been developed in Europe and internationally, some of which are commercial products and some research tools. However a major drawback of these tools is that they tend to be closed and difficult for other interested parties to extend making it difficult for the work of a larger research community to be combined. The HOL system is quite open but it lacks the strong development methodology provided by approaches such as B and VDM. The RODIN open tools kernel will greatly extend the state of the art in formal methods tools allowing other parties to integrate their tools, such as model checkers and theorem provers, as plug-ins to support RODIN methods. This is likely to have a significant impact on future research in formal methods tools and will encourage greater industrial uptake of these tools.

The use of UML and the MDA approach is gaining widespread use in industry. However UML lacks a coherent formal basis and formal analysis tools, while MDA lacks a formal basis for relating models at different levels of abstraction. RODIN will contribute to a formal basis for UML and MDA as well as analysis techniques and tools that can support their rigorous application.

3. Participant list

List of Participants

Partic. Role*	Partic. no.	Participant name	Participant short name	Country	Date enter project**	Date exit project**
CO	1	University of Newcastle upon Tyne	UNEW	UK	1	
CR	2	Aabo Akademi University	AAU	Finland	1	
CR	3	ClearSy System Engineering	ClearSy	France	1	
CR	4	Nokia Corporation	Nokia	Finland	1	
CR	5	Praxis Critical Systems Ltd	Praxis	UK	1	
CR	6	VT Engine Controls Ltd	VTEC	UK	1	
CR	7	Swiss Federal Institute, Institute of Technology Zurich	ETHZ	CH	1	
CR	8	University of Southampton	UoS	UK	1	

*CO = Coordinator
CR = Contractor

Since the proposal was submitted the consortium has decided to replace one academic partner (Institut National Polytechnique de Grenoble, France) with another (ETH, Zurich, Switzerland). The main reason for this was recent move of J.R. Abrial to ETH, where he was offered a position of Professor. This replacement has clearly made our consortium stronger and makes it easier for us to deliver the project results. First of all, Prof. Abrial is now located full time at the side of a ROIDN partner, where he is building a strong research group. Secondly, ETH is a world-class research institution and the project will tremendously benefit from its expertise by involving several staff members.

In the rest of the document the participants are referred to by the following names: Newcastle, Aabo, ClearSy, Nokia, Praxis, VTEC, Southampton, and ETH. (Except where tables are pasted from CPF documents where “official” short names will be left in.

4. Relevance to the objectives of the specific programme

RODIN is a project that aims to create a methodology and supporting open tool platform for the cost effective rigorous development of dependable complex software systems and services. Its sharply focussed objectives fall squarely within the remit of the strategic objective 2.3.2.3 *Open development Platforms for software and services* of the FP6 second call.

RODIN is inspired by the focus of IST in FP6 on *the future generation of technologies in which computers and networks will be integrated into the everyday environment, rendering accessible a multitude of services and applications through easy-to-use human interfaces*. It envisions that one of the main societal and economic challenges which need to be addressed when a multitude of services and applications is being rendered accessible is *solving “trust and confidence” problems so as to improve dependability of technologies,*

infrastructures and applications. Notably, producing methodologies and tools for improving dependability of complex software systems and services is the specific focus of this STREP proposal. It will therefore aim at making a significant contribution to one of the main technological objectives of IST in FP6 concentrated on *developing (...) software and computing technologies that are reliable, pervasive, interoperable and can be adapted to accommodate new applications and services*.

The project's central aim is to make significant contribution to the building of *open development (...) environments for software and services providing the next generation of methodologies, (...) and tools to support developers*. The related foundational research will focus on *fundamental design concepts, systematisation of domain specifications, concurrency, distribution, formal analysis and testing tools*. A key feature of RODIN's approach is a carefully balanced combination of expertise and knowledge contributed by industrial end-users, software developers, and academics, each contributing in a unique and strong way towards reaching the project's aims and objectives. Right from the beginning RODIN has been conceived as a project where *strong industrial users join forces with software and service suppliers in building common platforms with support of academic research partners*.

RODIN addresses in a truly comprehensive manner most of the key requirements of the Strategic Objective 2.3.2.3, as summarised below.

- *High level methods and concepts (esp. at requirements and architectural level) for system design, development and integration, addressing non-functional aspects, complexity, autonomy and composability*.

  To successfully tackle the growth in the complexity of software systems, RODIN adopts the view that the required solution must involve formal methods, aiming at design techniques that support clear thinking, clean architectures and rigorous validation and verification. There are two main forms of complexity the project will focus on: the architectural complexity which will be addressed using some well established formal techniques, such as abstraction, refinement and decomposition, and the complexity caused by the environment, which will be addressed through adopting a *systems approach* to software development. We will deal with the relevant high level methods and concepts by focussing on the following aspects:
  - formal representations of architectural design, decomposition and mapping principles (in Task 2.1, we will work on an integration of model checking with refinement and decomposition, in particular, aiming at integrating them into the industrial settings, and investigate how formal development and verification by refinement and model checking can enhance model-driven approaches to software engineering, such as MDA);
  - reusability (in Task 2.2, we will experiment with two, dual and complementary, approaches to reusability, viz. through generic instantiation and refinement);
  - development templates for fault-tolerant design methods (in Task 2.3, we will propose a number of well-documented templates assisting developers in formal specification and refinement of fault tolerance in the entire development process);
  - development templates for reconfigurability, adaptability and mobility (in Task 2.4, we will identify general abstractions expressing adaptivity, reconfiguration

and mobility, and investigate decomposition and mapping principles supporting stepwise development of open systems);
- as well as requirements evolution and traceability (in Task 2.5, we will develop an approach to traceability based on adding more structure to informal requirements documents).

- *Open and modular development environments, enabling flexibility and extensibility with new or sector-specific tools, supporting different adaptable development processes and methodologies and ensuring consistency and traceability across the development lifecycle.*

  Open platforms, middleware and languages supporting standards for interoperability, composability and integration.

  RODIN will deliver a development environment which will consist of two major collections of integrated tools, viz. an open tool kernel supporting extensibility of the underlying formalism and integration of tool plug-ins, and a set of actual plug-ins for a number of key areas relevant for dealing with the development of complex software systems. The development of the tool kernel will involve both an adaptation of the existing tools (in Task 3.2, we will do it for static analyzers; in Task 3.3, for proof obligation generators; and, in Task 3.4, for provers), and the design and implementation of new ones (in Task 3.6, we will work on the Project Manager tool; and, in Task 3.7, we will provide the Open Platform in which the various tools will be integrated). The flexibility and extensibility of the resulting tool kernel will be supported by a careful definition of the underlying formal technique (in Task 3.1, we will finalize Event-B so that the user will be supported by an adaptive re-use of generic models in the development process), and by the design and implementation of the Connection language (in Task 3.2, we will provide a support for smooth integration of various plug-ins). The development of plug-ins will result in a range of tools supporting the application of the RODIN methodology. The specific areas covered by these tools are as follows:
  - linking UML and B (in Task 4.1, we will work on improved representations of the following key concepts in UML: fault tolerance, component reuse and MDA, through extending the existing U2B tool);
  - Petri-net based model checking (in Task 4.2, we will extend the existing model-checking technique based on net unfoldings to systems with mobility - it should be stressed that model checking should be seen as complementary to the deductive proof techniques supported by the open tools kernel);
  - constraint-based model checking and animation for B (in Task 4.3, we will extended the existing prototype animator model checker ProB to deal with automated checking of refinement between models);
  - model-based testing (in Task 4.4, we will work on model-based testing which involves the automated generation of test cases from a formal model of a system that can be used to test the implementation of that model); and
  - code generation (in Task 4.5, we will work on an automatic translation of verified Event-B designs into one of classical programming languages, resulting in programs which are correct by construction).

Moreover, some case studies (such a mobile communications and ambient campus in WP1) will incorporate the use of standard middleware, and require interoperability of services and devices.

RODIN has made extensive provisions for transferring the solutions we develop into business and technological assets. From this perspective, the presence of ClearSy and Praxis, who are both software developers and service providers, is crucial. During our progress towards achieving the RODIN vision, we also plan to establish and develop partnerships with various industrial companies. This will take the form of an Industrial Interest Group, whose members are interested in the integration of formal methods in their software lifecycle. Moreover, as part of Task 5.5, ClearSy – directly and, possibly, through a dedicated subsidiary – will market the resulting platform as a systems engineering product.

RODIN will also promote technology transfer to SMEs in the following two main ways. First, it will *seek to build partnerships* with SMEs through the dedicated mechanism of Industrial Interest Group. Second, by opening the platform, the project will encourage SMEs to use it as well as to extend it with plug-ins specific to their areas of expertise, and to build their business around it.

In all its objectives, the project builds on areas in which Europe and the project partners already take an international lead. RODIN contributes to the IST aim *to increase innovation and competitiveness in European businesses and industry and to contribute to greater benefits for all European citizens* by enabling European industry to develop, implement and maintain software systems of ever growing applicability and complexity. This will be achieved through the provision of methods and open development tools for tackling the two main forms of complexity which will always be obstacles in the development of software systems. It will also *enhance and complement work in software initiatives at member state level*, in particular, by having the tool development aspect when compared to DIRC (Dependability Interdisciplinary Research Centre, Newcastle University, UK) and introducing openness when compared with the CREST (Center for Reliable Software Technology, Aabo Akademi University and Turku Centre for Computer Science, Finland).

5. Potential Impact

The cost, to EU commercial and government organisations, of creating and purchasing software is enormous. Precise figures are difficult to obtain because much of the effort is "in company software production" rather than just purchased software. Estimates were made in connection with the US PITAC [PITAC 1999] report that there are about 2 million people involved in software creation in the US. Because of the variety of regulations and languages, it is likely that a greater number of highly paid professionals are involved in some form of programming within the EU. To this huge cost must be added the money spent on purchasing software products and bespoke systems built by non-EU contractors. It is therefore clear that software creation is a major issue for the EU economy. It is however likely that this cost is actually exceeded by the loss of productivity resulting from poor software. (One major concern here is relatively poor "fault tolerance" behaviour of systems.) UK reports have observed that an organisation might spend 500€ on a PC for a professional, 1000€ on software but then waste several thousand Euros per

year of that professional's time because of the difficulty of using poorly designed systems. With major complex systems, the situation is at its worst: in a hospital environment, the staff involved will include busy and highly paid professional doctors whose real task is curing sick people; in highly competitive areas like air transport, being able to optimise processes could make or break a company; in critical infrastructure, the ability to control vulnerabilities could reduce the risks of "CyberTerrorism". Ease of use, fault-tolerance and dependability are crucial but hard to achieve.

Europe is fortunate in that it has "strong academic research" in both Dependability and Formal Methods. The participants in the RODIN consortium bring these strands together and will employ them to build better software systems. This will have a major positive impact on European competitiveness in services and embedded systems through productivity and quality gains. (The UK "Computing Research Committee" UKCRC has identified Dependable Software as a "Grand Challenge" for researchers.) The RODIN project brings together exceptionally strong researchers in both formal methods and dependability; these scientists have a strong track record of seeing their research through to practical application. The RODIN project will provide a base on which research will be reinvigorated and its application extended.

The principal intellectual tool to handle complexity is *Abstraction*; the use of *models* is universal in engineering. To be useful, models must be formally based so that they can be reasoned about both to draw conclusions about a model at one level ("horizontal" reasoning) and to check that models at different levels are correctly related ("vertical" reasoning). The RODIN group is particularly focussed on reasoning about the ability of systems to tolerate faults. In all cases, *rigorous* reasoning which uses engineering judgement at the required level of formality is as important an aim as completely formal methods; but the former relies on the basis of the latter. Neither pictorial notations with no formal basis nor mono-lingual notations which force coding decisions into the early stages of design will work for large systems. Carefully judged lightweight rigorous methods can –if underpinned by sound formal models and supported by tools– provide enormous improvements in cost-effective development of dependable systems.

The RODIN project will make its impact along three different axes. Most fundamentally, new formally-based development methods will be created. Partners in RODIN have unequalled track records in formal methods *and* in seeing them through to industrial use. The new objective is to create a set of methods that specifically addresses fault-tolerant systems (other members of the consortium have equally distinguished track records in research on Dependability). The project itself will use the created methods on a series of case studies within the RODIN project both to check and hone the methods. The impact of these case studies will be their role as demonstrators and as aids to technology transfer of RODIN methods. The most immediate external impact will be the tools platform created by the RODIN project.

The RODIN group is making a major commitment to *open source delivery* by providing a key tool framework as open source code. More significantly, the RODIN project will also deliver relevant specifications with the source code. (This has been done by few groups but was achieved by Jones' group for the "mural" theorem proving assistant – see [Jones et al 1991].) Abrial has designed and built the definitive tools for B and is a leading expert on the requirements which make such tools usable in industry. Butler has built tools which link B with the widely used UML notation.

The availability of a cleanly architected tools framework with specifications and code will provide a platform for other formal methods and dependability tools in Europe thus reinvigorating research in areas where Europeans were the initial contributors but have fallen behind the US in some tool support areas.

The RODIN consortium plans to populate the open-source tools platform with several collections of illustrative tools to support the development of dependable complex systems. These include

- support for a new version of Event-B;
- model checking tools;
- tools which link Event-B to UML;
- tools to support formal design using fault-tolerant constructs; and
- tools to support reasoning about faults as interference.

In addition to these tools, full publication of scientific results is planned (members of the project have extensive publications record in the related fields [Jones 1980, Abrial et al 1980, Jones 1990, Abrial 1996, Randell 2000, Xu et al 2002, Jones et al 2002, Butler and Falampin 2002, Petre et al 2002, Gaudel et al 2003]). The created tools will be tuned by exposure to the case studies and the collaborators Nokia, Praxis and VTEC are to be involved in architectural discussions from the beginning.

Of crucial importance is the early availability of the Praxis CDIS example [Hall 1996]. CDIS Air Traffic Control Display system was a major system whose development used formal notations (mainly VDM and CCS); its availability on "day one" of the RODIN project will ensure that all new ideas are tested against a real application.

The freely available (specified) open source code for the tools platform will serve as a focus for wider dissemination far beyond the three year funding of RODIN. Furthermore, the team's commitment to open-source specification will be a spur for open-source specifications of key *standards* such as those from OGSI. Public scrutiny of the specifications of major systems is today possible but needs both tool support and a firm series of examples to make it widespread.

Although several of the partners have been active in such tools projects before, they consider that no one European country could provide all of the skills required to create the framework which the RODIN project envisages. The differing backgrounds of the partners will provide a constructive engineering tension to ensure a more widely usable framework than could be built by any of the separate partners. Above all, the synergy between formal methods and dependability researchers will be key to a successful architecture.

Project work will also contribute to rigorous development of complex mobile systems. The partners have strong backgrounds in relevant formal methods (notably the π-calculus) and in developing advanced fault tolerance techniques suitable for such systems, and intend to combine their efforts using Nokia's case study on mobile telecom service and the "ambient campus" case study as the main drivers.

5.1 Standards Activities

The consortium will enhance its existing contacts with standardization activities around UML, OGSI etc. In particular, Nokia is now involved in standardization at ETSI of a UML profile for communication systems. Via project involvement into the engine failure management system case study provided by VTEC the group will be seeking to contribute to development of the next version of D-12B (RTCA Do178/B) - the standard for airborne civil aviation equipment software. The previous standards track record of the formal work is excellent: VDM and Z are the only two formal methods to have reached full ISO standard level.

5.2 Contribution to policy developments

Partners' participation in other European and national projects will be used to disseminate the project results. The RODIN partners are involved in a number of national projects, notably

- Newcastle leads the UK "Dependability IRC" which is a six year project funded at more than 12M€ which is concerned with Dependability of wider "Computer-Based" systems
- The researched to be conducted by Aabo is carried out within the *Center for Reliable Software Technology* (CREST). For the years 2002–2007, the CREST research groups have been nominated by the Academy of Finland as a Centre of Excellence for Formal Methods in Programming.
- The Aabo group leads both the FOSSE (Formal System and Software Engineering) and TORES (TOols for REliable Software construction) research projects funded by Finish sources.

The project will run an *Industrial Interest Group (IIG)* which will both provide feedback on RODIN's evolving plans and will serve as a technology transfer pathway. Companies who have already signed up to IIG membership include IBM UK, Adelard (UK), Gemplus (France), I.C.C.C. (Czech Republic), STMicroelectronics (France), DGA (France), but it is our intention to promote and widen IIG membership throughout the project with a particular focus on involving SMEs.
This is a brief overview of the current IIG members:

- The Model-Driven Business Integration team at IBM UK Labs in Hursley have expressed strong support for the research proposed in RODIN. The IBM team develop techniques and tools supporting MDA standards for application to business systems integration. They see the formal approaches being proposed by RODIN as providing a more rigorous basis for the correctness of MDA transformations. The IBM team make extensive use of the Eclipse platform and will be a valuable source of advice on Eclipse to RODIN.
- Adelard is a small but influential dependability and safety consultancy (http://www.adelard.co.uk/). It is extremely active on both main topics of RODIN's research: formal methods and dependability. It is the company's view that the project plans are realistic and could offer an excellent chance for European researchers to recapture the lead in these key areas. Adelard will be aiming to

apply RODIN results in two particular areas: applied research in safety and software reliability and software development and analysis, including formal methods and static analysis.

- Gemplus is the world's leading provider of smart card solutions. The R&D department is in charge of designing new tools and methods to support building of higher EAL compliant devices, for which formal methods are required. RODIN would provide a platform for the development of such devices, integrating all the tools required for the security demonstration and providing extension points for the development of dedicated features.
- I.C.C.C. Group is a leading supplier of IT solutions. It has an extensive experience in software development and open platforms for development of tools integration, 3D representations in industry and entertainment, wireless technologies and mobile applications, security and data transmissions, and is involved now in IST project Ophelia aiming to build a open platform and methodologies for development tools integration in a distributed environment, The company plans to use RODIN's results to make their platform and methodologies more rigorous.
- DGA (Direction Générale pour de l'Armement) is a military governmental organization integrated into the French Defence Ministry. Complex Systems Engineering department is in charge of R&D activities centred on tools, means and techniques required to improve complex systems engineering methods. It is expected that RODIN will contribute to their activities by bringing a complete framework usable for the design of military complex systems.
- STMicroelectronics is one of the six major microelectronics companies in the world. Its smartcard division is in charge of the complete design of smartcard devices, including software and System on Chip development. ST lacks means for structuring in house developments at the system level. It is the company's intention to use their participation in IIG to bring system level design capabilities as well as promising features like IP reuse and correct by construction product.

Australian national research centre in Information and Communications Technology (NICTA http://nicta.com.au) have expressed a very strong interest in exploring opportunities to collaborate with the RODIN project, through mechanisms such as the Australian government's Innovation Access program, which funds Australian research projects linked to international projects, including EU 6th Framework projects. This is a clear example of how project open source delivery will lead to the intensive impact on software R&D.

6. Project management and exploitation/dissemination plans

6.1 Project management

The Project Management structure is influenced by the experience – and based on the best practice developed by its members – of many years of involvement in ESPRIT and other

European Programmes. In particular, Newcastle University has twelve years experience Coordinating around 15 major European collaborative projects and networks.

In the light of this experience, it is our intention to maintain as lightweight a management structure as possible within the constraints imposed by the need to deal effectively with intra- and inter-workpackage co-ordination and communication, and with the requirements detailed in the contract.

The project's infrastructure will use the Internet, and will consist of an internal and public web server, and of internal servers for documents, code and documentation, version control and archived mailing lists. The infrastructure of a previous project will be (re)used, as far as possible.

The interface with the Commission and overall responsibility for RODIN rests with the Project Coordinator. This will be Alexander Romanovsky from Newcastle. There will be two separate sub-domains of management and co-ordination within the project:

- The Executive Board that is responsible for the overall conduct of the project, and is the supreme decision-making body;
- The Administrator who is responsible for providing day-to-day managerial assistance to the project.

Membership of the Executive Board will consist of one representative of each Partner, though other individuals may be co-opted as appropriate. Where the Partner defines the role of principal investigator he or she will normally be the representative on the Executive Board. The Project Co-ordinator chairs the Executive Board and will delegate all administrative responsibilities to the Administrator (Jon Warwick, Newcastle). The Board will have the primary responsibility of ensuring effective technical collaboration in pursuit of the overall aims of the project.

The primary responsibilities of the Executive Board are as follows:

- to reach agreement between the Partners on relevant major technical issues
- to provide regular co-ordination and review of the technical workpackages
- to approve the major project deliverables prior to submission to the EC
- to organise major project internal workshops and project reviews
- to support the Administrative Co-ordinator by providing the necessary authority on non-technical matters on behalf of each institution
- to deal with any non-technical issues between the Partners that cannot be resolved at the operational level by the Administrative Co-ordinator
- to co-ordinate the technical work of the project with that of other relevant IST projects
- to promote knowledge and understanding in circles external to the project (for example in industry, other research institutions, and academia)
- to resolve any conflicts that may arise during the course of the Project, following the Conflict Resolution Procedure described below.

The Executive Board will meet at least twice a year, normally in connection with Project internal workshops. Additional meetings will be called as and when necessary. At each meeting of the Executive Board it will receive a report from the Administrator (who will act as secretary to the Board). Day-to-day communication amongst members on Executive

Board matters will normally be by means of an electronic bulletin board. As regards the day-to-day technical operation of the project, responsibility for the different components of the Workplan has been divided among the Partners. The organisation is as shown here:

- WP1 Research drivers (Aabo)
- WP2 Methodology (Newcastle)
- WP3 Open tool kernel (ClearSy)
- WP4 Modelling and verification plug-ins (Southampton)
- WP5 Dissemination and exploitation (ClearSy)
- WP6 Project management (Newcastle)
- WP7 Project review and assessment (Praxis).

For each work item, the workpackage leader is responsible for the completion of the work and the timely production of the deliverables. Workpackage leaders will report to the Executive Board. Newcastle will ensure the overall coordination of the Project.

The Administrator is primarily responsible for managing the budget; ensuring contractual deliverables are in fact delivered on time; ensuring the proper dissemination of internal and external technical (and other) reports; receiving from each site from time to time such technical, financial and administrative information as the Commission may require and forwarding it to them.

The Administrator, who will be based at Newcastle, will carry out the following duties in a timely manner, under the direction of the Executive Board (through the principal investigator at the co-ordinating contractor site):

- develop and maintain expert knowledge of contractual matters that are administrative and financial in nature
- collect, monitor and integrate administrative and financial information from Project partners
- ensure the collection, storage and distribution of Project information, technical as well as administrative and financial
- convene the meetings of the Executive Board
- record the decisions taken by the Executive Board and monitor their implementation
- ensure that information requested by the Commission is duly transmitted as and when required
- ensure the obligations of the Co-ordinator to the consortium as a whole are expedited (particularly with respect to financial and administrative matters)
- establish and maintain an efficient information service on the activities of the Project for use by members of the Project and interested external parties

Conflict Resolution Procedure: In normal operation, conflicts will be resolved in discussion with the appropriate workpackage leader or, in the case of non-technical matters, the Administrative Co-ordinator. Where a conflict is not amenable to resolution through such discussions, the matter will be raised at the next scheduled Executive Board meeting or, if requested by any one of the Partners to the Project, at a specially convened

meeting. Any conflict that is not amenable to resolution through discussion in Executive Board will be resolved by means of a vote. Each of the Partners to the Project will be eligible to cast a single vote. The decisions of the Executive Board, whether arrived at by discussion or vote, will be binding on all Partners to the Project. Finally, in addition to compliance with the standard model contract, it is our intention to produce a Consortium Agreement that will codify the specific rights and obligations of the Project Partners with respect to this particular project.

6.2 Plan for using and disseminating knowledge

Management of knowledge and intellectual properties will be dealt with by way of a Consortium Agreement. This document will be developed, signed and in force during the contract negotiation period. The process will be driven by experts at the Coordinating Institution (Newcastle) and the negotiation of this agreement will make use of appropriate expertise throughout the Consortium. Units such as the Newcastle University Technology Transfer Office will also be brought in when appropriate, to offer advice and relevant experience.

Major parts of the tool software developed in RODIN will be released as an open source, in particular, those parts of the tool kernel supporting interoperability and several of the plug-in tools. The appropriate licensing policy will be chosen and finalised during the consortium agreement negotiation. We will consider, among others, the GNU general public licence, CPL (Common Public Licence) and the BSD licence.

WP5 on Dissemination and Exploitation (see section 7.1.5) presents concrete plans and actions which RODIN partners plan to undertake in knowledge dissemination beyond the consortium (including creation of a project web site, organisation of scientific International workshops presentation of tutorials, preparation of educational materials, exploitation of Industry Interest Group, etc.). Newly-created university courses, educational materials and experience and expertise accumulated by the partners will play a crucial role in knowledge dissemination after the project lifetime.

6.3 Raising public participation and awareness

RODIN partners have been successful in raising public awareness in the relevant areas. For example, DIRC (led by Newcastle) activity in the area of ensuring secure and safe human-machine interaction in the context of cockpit has been recently publicised via BBC (TV and radio channels) and general public newspapers in the UK. Another example is DIRC participation in CeBIT 2003 (Hannover) which aimed at a widest possible dissemination of the research results to a non-specialised general audience. RODIN partners will be looking for similar opportunities to publicise and to spread awareness of their work.

7. Detailed Implementation Plan

7.1. Introduction – general description and milestones

The development of the design methods and the tools platform will be guided by and validated by five industrial case studies. The development of the methodology will be

achieved by enhancing and combining existing approaches to formal design and fault tolerance design. Our approach to providing the support platform will be to design an open kernel for the core language and proof methods. We will also provide mechanisms for language extensions and the integration of plug-ins. In addition we will provide a profile of integrated plug-ins supporting the methods. Integration will be achieved on a generic open platform such as Eclipse.

It is the intention of the RODIN team to be as generic as possible with respect to the formal methods supported. In particular:

- any theory developed (e.g. for event-based systems, rely/guarantee reasoning, atomicity refinement) will be described in scientific research papers as generally as possible
- the tools will be developed on top of a framework which is itself generic (but capable of instantiation to a range of methods)

But the very experienced team on RODIN are fully aware that one must have a specific deliverable to test and refine ideas.

To indicate that the referees' concern about "justifying" the choice of formal notation was misplaced, the following facts are rehearsed. Prof Jones was one of the small group which created the formal development method VDM; Prof Abrial was the initiator of the Z specification language; Prof Jones contributed key ideas such as data reification, the "logic of Partial Functions", rely/guarantee conditions to development methods. Prof Abrial has more recently developed the B method (which has interesting connections with VDM). Nor is the team's formal method strength limited to the older members of the team: key contributions have been made by both the Aabo and Southampton groups. The fact that the RODIN team recognises the need to choose a single B-based focus to demonstrate and test its ideas ought to be seen as a major strength of the consortium. We will ourselves, check the kernel can support a range of formalisms; we will encourage the outside organisations to build supports for their methods in our kernel. But the RODIN project itself must have a focus in order to prove its objectives to be met. The case for the selection of Event-Bis given below in section 7.1.3. RODIN will be structured as four technical *workpackages*: Research Drivers, Methodology, Open Tool Kernel, and Modelling and Verification Plug-ins. These workpackages are strongly interlinked. The Methodology workpackage will produce the main fundamental results on rigorous system development using the existing experience of the partners and material from the Research Drivers workpackage. The Research Drivers workpackage will feed back the Methodology workpackage to ensure applicability and usability of produced theoretical results. The results of the Methodology workpackage will be used in the Open Kernel workpackage and Modelling and Verification Plug-ins workpackage for developing the supporting tool platform. The Research Drivers workpackage will serve to validate the methods and tools and will provide guidance on how the methods can be integrated with existing industrial processes. There will be three further workpackages for Dissemination and Exploitation, for Project Management and for Project Review and Assessment.

Modelling and Verification Plug-ins and Open Tool Kernel workpackages although both connected with tool building, are split into two separate parts: the reason for this is essentially one connected with openness versus closeness. The Open Tool Kernel workpackage is concerned with the construction of the basic general kernel tools: syntax checker, proof obligation generator, and prover, which, once built, will remain quite stable

over the years. On the other hands, the Modelling and Verification Plug-ins workpackage is concerned with the plug-ins. These are the additional sub-tools which could be constructed for specific needs: an animator, a model-checker, a sequential program generator, a VHDL translator, a protocol generator, a connection tool between Event-B and UML, etc. All such tools will work with genuine output (proved Event-B) provided by the general kernel. By definition, the list of such sub-tools is open and, in fact, we shall only produce a small number of them within RODIN. Our aim is thus to have two distinct workpackages precisely to enforce this separation, and also clarify the design and final implementation of these categories of tools.

To demonstrate how the workplan will lead the consortium to achieving the objectives of the project, below we describe the work subsumed under each workpackage.

7.1.1 WP1 – Research drivers

The general objectives of this workpackage. This workpackage plays the central role in RODIN since it will

- drive the development of the methodology of WP2, validate it and evaluate its cost-effectiveness via the dedicated metrics established in WP7
- consolidate the efforts of partners in development of unified methodology
- provide the feedback to the open platform developed in WP3
- set requirements, guide the development and assess plug-ins developed in WP4

Description of work

The workpackage naturally emerges as a driver of research and innovation activities of RODIN. Indeed, it not only promotes industry-academia collaboration but also joins the efforts of partners involved in the development of the methodology and tool support. Therefore, the leader of this workpackage should understand all of the case studies, be actively involved in the development methodology and tool support and have a strong experience in industry-academia co-operation. Because Aabo satisfies all these criteria it leads this workpackage.

The workpackage consists of five case studies which belong to various application areas: control and embedded systems, mobile and ambient systems. The case studies, along with the responsible partner, are:

Case study 1: Formal Approaches to Protocol Engineering (Nokia)

Case study 2: Engine failure management system (VTEC)

Case study 3: Formal Techniques within an MDA Context (Nokia)

Case study 4: CDIS Air Traffic Control Display System (Praxis)

Case study 5: Ambient Campus (Newcastle)

It is important to note that the systems themselves will not be funded by RODIN; any EU funding is used to develop and refine RODIN methodology. These are systems that are already under development in the relevant partner organisations. The partners leading the case studies will provide material (requirements, specifications, designs) that will guide and validate the research in RODIN. Industrial partners will apply the RODIN methods to their case studies, developing formal models and refinements, supported by the academic partners. This is why the Project work on WP1 cannot be classified as demonstration activity.

The research within the workpackage will proceed in a cyclic way. We foresee several cycles for each case study. The essential steps of the cycle are as follows:

1. Industrial partners: problem identification
2. Academic partners: conceptualization of the problem
3. Industrial and academic partners: joint efforts to create methods and tools assisting in solving the problem
4. Industrial partners: approbation of the developed methods and tools and feedback on their validity.

Each cycle will allow us to refine the methodology and supporting tools to adjust them to the real industrial development processes. Therefore, our research in this workpackage will *concurrently solve the problems of real systems development and generate new knowledge*. Diversity of case studies will facilitate generalization of the research results and support development of unified methodology.

Each of the case studies will focus on formal approaches to tackling the acute problems of the domain areas. The industrial partners have identified these problems together with offering the systems which they are developing or have developed as test benches. Although the systems belong to diverse application areas they have a common stringent requirement – to provide a high degree of dependability. Each of the case study developments will perform tasks in common phases as follows:

- Initial definition and detailed objectives of the case studies.
- Initial development of formal models following RODIN methodological approach.
- Application of the RODIN prototype tools to the initial developments.
- Further development including consideration of issues of integration and reuse.
- Evaluation of the RODIN methods and tools in the context of the case studies.

Our case studies should drive us towards our main goal – creating a *viable* cost-effective methodology and a comprehensive tool platform for development of complex dependable systems. To ensure this the workpackage is formed over a representative but non-redundant subset of systems and development processes. Below we demonstrate the selection criteria for the case studies:

Criterion	CS1	CS2	CS3	CS4	CS5
Maturity level of organizations work on formal methods	Advanced in model checking	Entry level	Average in refinement-based	Advanced in refinement-based	Advanced in model checking and refinement
Type of application	Mobile system	Safety-critical control system	Mobile system: architectural view	Safety-critical information system	Ambient system
Complexity	Very complex	Average	Complex	Average	Very complex
Development stage	Developed	Developed	Initial	Developed	Initial
Fault tolerance will focus on	Design faults	Hardware fault	Architectural errors	Human errors and design faults	Design faults, software faults
Main methodological focus	Integration between refinement and model-checking	Ensuring safety via formal reasoning about fault tolerance	Formal methods for agile develop-ment: towards formalization of MDA	Viability of RODIN methodology	Refinement and model-checking of ambient systems
Main tool focus	Model-checking plug-ins	Linking UML and B	Linking UML and B, model –based testing	Linking UML and B	Model-checking, code generation

The chosen case studies create positive tension between different strands of our research and productive environment. Each plug-in and advances in each research direction contributing to RODIN methodology will be evaluated by at least two case studies.

We will present each case study in more detail by briefly describing the system, the tasks and the expected results of the case study. Each case study has its own list of objectives provided by the relevant industrial partner and these objectives contribute to the overall objectives of RODIN.

Case study 1: Formal Approaches to Protocol Engineering

The “Lyra” method developed within the protocol engineering group at Nokia supports the model-driven approach to protocol engineering. The method covers all stages of the development. The development method supports service-oriented, i.e. requirement-based, 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. Currently algorithmic verification is 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 model checking is prone to the state explosion problem. Nokia will investigate the use of refinement to proof decomposition steps. Hence the important problem to be tackled within the case study is a combination of top-down (refinement) and bottom-up (model checking) verification techniques in the development of communicating systems and telecommunication protocols. Also the applicability of techniques for formal reasoning about fault tolerance in this application area is to be investigated within the case study.

The case study will be centred 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 the UMTS radio access network. PCAP is specified to manage the communication between the network elements *Radio Network Controller (RNC)* and *Stand-alone Assisted Global Positioning System Serving Mobile Location Center (*SAS).

The objectives of this case study are to:

- investigate the use of refinement and model checking in model-driven approach to development of communicating systems and protocol engineering,
- investigate benefits of using refinement approach versus algorithmic verification to verify system decomposition and composition,
- investigate model reduction techniques and proof methods for data abstractions,
- investigate use of model checking to verify correctness of system components,
- investigate applicability of formal reasoning techniques about fault tolerance in this application area,
- validate top-down and bottom-up formal techniques and supporting tools by using them in the development of a *Position Calculation Application Part (PCAP)* specified by the *Third Generation Partnership Project (3GPP)*.

Tasks

T1.1.1 Define the case study, evaluation plan, measurements and assessment criteria

T1.1.2 Review the “Lyra” method and identify the development steps, which should be tackled by RODIN. Assign the sets of methods and techniques to be applied at these steps.

T1.1.3 Develop UML models of PCAP according to “Lyra” method. Validate the Service Specification model(s) and some chosen system components using model checking and refinement techniques. Investigate model reduction techniques and proof methods for data abstractions.

T1.1.4 Investigate use of refinement and model checking to verify decomposition and composition steps. Investigate the combination of model checking and refinement techniques in context of UML and B. Investigate the use of model checking tools in

combination with UML to B tool. Investigate the applicability of formal reasoning about fault tolerance in this application area.

T1.1.5 Investigate how formal and semiformal design techniques can be combined and used to support integration with the targeted platform. Use the developed plug-ins to tackle various problems of the development. Validate usability of the open platform.

T1.1.6 Evaluate benefits of formal techniques and tools applied within the case study.

T1.1.7 Write final evaluation report

Expected results

- Feasibility of incorporating refinement to verify model decomposition.
- Guidance on combining the model checking with the refinement approach.
- Enhancement of UML to B tools to support interplay between formal and semiformal methods, and the enhancement of model checking tools correspondingly.
- Validation of developed open platform and plug-ins.

Case study 2: Engine failure management system

The engine failure management system is based on a typical engine management system used in aerospace and is defined by VTEC. The primary function of the system is to manage the various failure modes of an engine control system.

The objectives of this case study are to:

- investigate the use of formal methods on engine failure management applications,
- investigate methods of using formal methods efficiently via reuse
- investigate methods for specifying a generic software support package for engine failure management applications
- produce prototype products that could be used to support the development of a software support package for engine failure management applications
- investigate benefits of integrating UML and B.

Description of case study and tasks:

T1.2.1 Define case study, evaluation plan, measurements and assessment criteria.

T1.2.2 Produce an informal specification of a typical engine failure management system based on knowledge of existing products.

T1.2.3 Use the informal specification to produce a visual, object-oriented formal specification of a typical engine failure management system.

T1.2.4 Validate and verify formal specification to evaluate usefulness of formal methods.

T1.2.5 Investigate generalisation and re-use techniques using the object-oriented formal specification. Using these techniques, produce generic specification products from which other specific applications can be derived.

T1.2.6 Utilise generic specification products to obtain a typical example application. Evaluate the re-usability of the generic specification products.

T1.2.7 Write final evaluation reports

Expected Results

- Feasibility of formal methods in application domain including assessment of usability and benefits of object-oriented style of formal specification and rigorous validation and verification.
- Development of techniques for re-using formal methods within application domain.
- Enhancement of UML to B tools to support use of object-oriented formal specification within application domain and to support re-use features.
- Basis of a formally specified and developed generic package for supporting the development of specific failure management subsystems.

Case study 3. Formal Techniques within an MDA Context

Nokia has much experience in utilising an MDA style approach in the development of systems targeted at mobile and telecom systems. A model driven approach is especially useful for development since it allows for continual mapping of a platform independent model to platform specific models through model transformation. However, such issues as the form and semantics of the model transformations as well as selecting optimal transformation for each stage of the development must be studied further. Nokia is interested in applying formal techniques such as refinement to ensure that the models remain consistent to high-level requirements.

An introduction of a completely formal development requires redesign of the entire business process which is considered to be too risky. A pure formal approach has been shown not to work with exceptionally large, complex systems and requires large changes to the processes already in place. Many of Nokia's systems would not normally be considered safety critical or be amenable to formal approaches. However light weight or **pragmatic** use of formal methods has been shown to be greatly beneficial, especially in areas such as architecture and protocols. Hence Nokia is interested in identifying and validating a set of **formal techniques** which support the model driven approach and can be integrated into MDA style development through the construction of a suitable modelling (or meta-method) framework.

The objectives of this case study are to:

- Investigate how formal techniques fit into the Model Driven Architecture (OMG MDA) framework as "MDA Mappings"
- Investigate which techniques are applicable at which stages of platform independence and platform specific models.
- Investigate how to integrate and compare the verification and validation results from the various levels of abstraction
- Investigate methodological issues relating to formal model development with an emphasis of refinement and retrenchment

From the point of view of the case study material we will be concentrating on initially the MITA (Mobile Internet Technical Architecture) framework. RODIN will:

- Investigate formal specification of a high-level architecture such as MITA
- Investigate formal specification with respect to meta-modelling (architecture becomes the language in which the models are developed)
- Investigate the PIM to PSM (platform independent model, platform specific model) mappings of an architecture such as MITA

Later integration of these MITA specifications with the other Nokia case study will be made. This is an extension of the work attempted in the earlier PUSSEE project.

Tasks

T1.3.1 Define case study, evaluation plan, measurements and assessment criteria

T1.3.2. Produce an informal specification of a relevant subset of MITA. Define the development steps to be undertaken in MITA framework, assign the sets of methods and techniques to be applied at different development stages.

T1.3.3. On the basis of informal specification develop models in suitable representations (nominally UML with a more concrete semantics) and apply formal techniques for validation and verification.

T1.3.4. Investigate the effect of requirements changes on informal and formal system models. Investigate benefits of formal techniques to tracing changing UML models.

T1.3.5. Undertake further development steps, primarily integration of other case studies (when applicable) with the MITA framework.

T1.3.6. Evaluate benefits of formal techniques and tools applied at different stages of the development.

T1.3.7. Write final evaluation report

Expected results

- Feasibility of incorporating formal methods with MDA
- Support for modelling framework
- Guidance on usage of formal techniques in MDA development
- Enhancement of tools to support formal development with MDA
- Enhancement of supporting languages (notations, semantics etc) to support the types of engineering practice in use at Nokia
- Validation of the RODIN platform

Case study 4: CDIS Air Traffic Control Display System

CDIS is a safety-related system that is responsible for displaying information to air traffic controllers 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 objectives of this case study are to:

- provide some of the initial drivers for the theoretical work in RODIN, by offering specification, design and verification material from an existing industrial-scale

development and identifying the challenging issues that arose during that development

- exercise the intermediate tools deliverables of the RODIN project by subjecting them to the "stress testing" that only real industrial problems can provide
- provide a realistic assessment of the final results of RODIN by comparing what was possible in the 1990s with what can be achieved on the same development with the RODIN notations and tools

Tasks:

T1.4.1 Identify a subset of the CDIS formal specification and design documents that still define a coherent system, that retain the diversity of notations and techniques that were used to specify and design the system, and yet which is small enough to be manageable. Ensure that the scope of the CDIS subset covers a significant proportion of the RODIN research areas.

T1.4.2 Construct new specification and design documents that define the CDIS subset identified in Task **T1.4.1** and add annotations to these documents that identify the challenging issues faced by the developers that had to be resolved only informally in the original development. These issues include (but are not limited to) the satisfaction of the system's availability requirements and the identification, implementation and verification of error-recovery mechanisms.

T1.4.3 Circulate the new specification and design documents within the RODIN project and respond to questions concerning the identified challenging issues.

T1.4.4 Rework the specification and design documents to use the specification and design notations nominated by the RODIN project (including Event-B and UML).

T1.4.5 Use the intermediate RODIN toolset to apply syntax and type-checking to the formal specification and design, to perform refinement proofs, and to apply model-checking to the CDIS protocol specification.

T1.4.6 Write final evaluation reports. This task will include a detailed assessment of the advantages provided by the results of the RODIN work over what was possible in the industrial development of complex systems 10 years ago. The CDIS case study thus acts partly as a reference model for RODIN.

Expected results

- Injection of real industrial requirements into the RODIN research effort at an early stage in the project.
- Validation of the RODIN model-checking approach on an industrial-scale concurrency specification.
- Validation of the RODIN refinement approach on an industrial-scale "model-based" specification and complex design.
- "Stress-testing" of the RODIN tools platform plug-ins in order to ensure they will be useful in an industrial environment.
- An industrial view of the success of RODIN, by providing a detailed comparison of the difference between what was achievable 10 years ago in the industrial development of complex systems and what will be achievable by the end of the RODIN project.

Case study 5: Ambient Campus

Ambient Intelligence (AmI) is an emerging field which unlike traditional technologies is lacking well-defined engineering methods. However, AmI applications are becoming typical for organizations distributed over a large area such university campuses, hospitals, factories, etc. Often these applications have high dependability requirements imposed on them. Software developed for AmI applications need to operate in an unstable environment susceptible to various errors and unexpected changes. Hence, it should be fault tolerant, adaptable and reconfigurable. In this case study we will investigate how to use formal methods combined with fault tolerance techniques to developed highly dependable AmI applications. Moreover, we will develop modelling and design templates for adaptable and reconfigurable software.

Ambient Campus case study will be based on the SRIF2 - Adaptive Computing Schools of EECE & CS Equipment/Resource Grant won by University of Newcastle in 2003 (total around €1.3M). This includes support (€0.7M) for building and researching ambient intelligence applications in the university campus (30 PDAs, software development tools, wireless network, etc.). The case study covers the development of several working ambient applications supporting various educational and research activities.

The objectives of this case study are to

- elucidate the specific fault tolerance and modelling techniques appropriate for the application domain,
- validate the methodology developed in WP2 and the model checking plug-in for verification based on partial-order reductions,
- to document experience in forms of guidelines and fault tolerance templates

Tasks

T1.5.1. Define case study, evaluation plan, measurements and assessment criteria

T1.5.2. Produce informal specification of the ambient campus; identify problems related to provision of fault tolerance

T1.5.3. Identify general solutions from the area of application level fault tolerance (such as atomic actions and exception handling) to be adapted to AmI applications. Apply adapted techniques to the ambient campus system.

T1.5.4. Investigate use of refinement-based approach to develop a chosen part of the system. Investigate problems specific to model checking –based verification of ambient applications.

T1.5.5. Identify a design abstractions to be used for rigorous development of fault tolerant applications, which are inherently distributed, asynchronous and mobile. Use the developed model checking plug-ins to verify correctness of the system

T1.5.6. Document templates for provision fault tolerance in AmI systems. Prepare guidelines for systematic application of fault tolerance during development of systems in the domain Validate usability of the open platform

T1.5.7. Write final evaluation report

Expected results

- Provision of guidelines on using formal methods and fault tolerance techniques to the ambient intelligence applications
- Validation of the developed platform and plug-ins

The contribution of the tasks in the individual case studies to the WP1 deliverables is described in the following table:

Deliverables	Case study 1	Case study 2	Case study 3	Case study 4	Case study 5
D1.1	T1.1.1	T1.2.1	T1.3.1	T1.4.1	T1.5.1
D1.2	T1.1.2	T1.2.2	T1.3.2	T1.4.2, T1.4.3	T1.5.2
D1.3	T1.1.3	T1.2.3	T1.3.3, T1.3.4	T1.4.4	T1.5.3
D1.4	T1.1.4	T1.2.4	T1.3.5	T1.4.4	T1.5.4
D1.5	T1.1.5	T1.2.5	T1.3.5	T1.4.6	T1.5.5
D1.6	T1.1.3, T1.1.4, T1.1.5, T1.1.6	T1.2.7, T1.2.4, T1.2.5	T1.3.5, T1.3.6	T1.4.5	T1.5.5, T1.5.6
D1.7	T1.1.7	T1.2.6, T1.2.7	T1.3.7	T1.4.5	T1.5.6, T1.5.7

7.1.2 WP2 – Methodology

Objectives. This workpackage will produce the RODIN methodology for rigorous development of complex systems. This will be done by making advances in the basic research areas related to system modelling and mapping of models, software reuse, and formal reasoning about system fault tolerance, reconfiguration, mobility and adaptivity. The methodology development will be driven by the industrial case studies (WP1) and by the existing experience in application formal methods accumulated by the industrial and academic partners. The methodology will be evaluated by the case studies. The results of the workpackage will further feed development of the concrete modelling and verification plug-ins (WP4) and the kernel (WP3) to allow partners to produce an automatic support for the methodology. The work will result in developing a number of guidelines on using the methodology.

Description of Work

T2.1 Formal representations of architectural design, decomposition and mapping principles

The RODIN methodology will adopt a system approach to software development. Within the approach we formally model both software and its operational environment and use refinement and decomposition as main development techniques. Refinement approach allows us to gradually introduce implementation details into our initial specification, decompose the system into independent components and map these components on various architectures. Such an approach has been proposed by Back and Sere [Back and Sera 1996, Back et al 1996] to introduce modularization and parallelism in the sequential programs. Refinement and decomposition techniques have been further elaborated by Sere and Walden [Sere and Walden 2000] to derive distributed programs by refinement, by Butler et al. [Back et al 1996] to develop control systems, by Butler [Butler 1996] to design communicating systems and by Sere and Troubitsyna [Sere and Troubitsyna 1999] to develop safety-critical systems. Decomposition is also important for bottom-up model

checking approach to verification. An integration of model-checking with refinement and decomposition will allow us to better scale-up these formal techniques. Therefore, rather than developing new formalisms we aim at elaborating on the existing well-defined methods to integrate them into the industrial settings and use in the new application areas.

Model driven approaches to software engineering (e.g., Model Driven Architecture – MDA) gain increasing popularity in industry. Like refinement these approaches adapt the top-down development paradigm. They support, albeit informally, incremental development by model refinement and decomposition. For instance, abstract modeling allows the specification of system functionality separate from the specification of the implementation of that functionality on a specific technology platform and hence allows platform independent models to be transformed to various target platform implementations. Obviously, dependability of models is crucial for model driven approaches. We are planning to investigate how formal development and verification by refinement and model checking can enhance model-driven approaches. We will extend our work on connecting UML to B, integration of formal reasoning in model-driven approaches as well as our ongoing work on the causality and concurrency in the π-calculus

This work will be closely related to all five case studies discussed in section 7.1.1.
The main novelty of this task is in formalizing model driven development with combination of refinement-based and model-checking techniques.

T2.2 Reusability, genericity, refinement

Reusability is a concept strongly related with those of *generic instantiation* and *refinement*. The latter are very complementary. A *generic development* is one in which some set constants (but also more concrete ones) in it are acting as *parameters* which can be instantiated. By doing such an instantiation we recognize that a certain *problem* (the instantiated one) has got some similarity with another (the generic one). As a consequence, a proved solution development for the latter can be entirely instantiated to construct the former. The interesting aspect of this approach is that the proofs of the generic development are thus instantiated "for free". A *refined model* is one in which the *solution* already elaborated in a more abstract model is now made more precise. Clearly both concepts could play some orthogonal roles in implementing reusability. For example, a generic development, made of a number of more and more refined models, can be partially generically instantiated. The outcome could then be refined several times and then again generically instantiated, and so on. In this task, we will experiment with this dual approach to reusability.

This task will extend the Event-B language and proof obligation generation so that developments (models, refinements, proofs) can be generic and generic developments can be instantiated to specific developments. It will develop guidelines on constructing generic system developments and instantiating them to more specific developments. The task will focus on reusing developments within rather than across domains that can be tailored and combined to construct a system for a specific market or customer. The domains in question will be those of the Engine Control, CDIS and Nokia's MDA case studies.
The idea of enhancing reuse via instantiating generic developments is novel and requires significant efforts to create such generic developments.

T2.3 Development templates for fault-tolerant design methods

There are two complementary approaches to achieve system dependability: fault avoidance and fault tolerance. Within RODIN such formal techniques as refinement and model checking are used to ensure correctness, i.e., to implement fault avoidance in the process of software development. However, to guarantee dependability of the overall system we need to build fault tolerant software, i.e., software which is not only fault free but also able to cope with faults of other system components. Obviously, this goal is attainable only if fault tolerance mechanisms are systematically introduced in the specifications and formally verified.

It is widely recognized that a high degree of dependability of computer-based systems can be achieved if dependability consideration starts from the early stages of system development. The focus of this task is on proposing a number of well-documented templates assisting developers in formal specification and refinement of fault tolerance in the entire development process. We will investigate mechanisms required to tolerate various types of faults of the environment (e.g., software design faults, faults of COTS, hardware random faults, human errors etc.). To address these types of faults we will integrate such fault tolerance techniques as atomic actions [Xu et al 2002], exception handling [Cristian 1995] and transaction compensations [Chessell et al 2002] in our system approach. This work will rely on our approach to formal specification and refinement of fail-safe fault tolerance mechanism [Troubitsyna 2003]. An acute issue of fault tolerance is a complete and adequate statement of fault assumptions. It has been demonstrated [Hayes et al 2003] that viewing errors as *interference* from an unwelcome concurrent process allows for earlier identification of environmental faults and measures for handling them. Also in our previous work we have demonstrated how to derive fault assumptions and error recovery procedures from safety analysis techniques and specify them formally [Sere and Troubitsyna 1999, Troubitsyna 2002]. In RODIN these approaches will be integrated to develop templates for the rigorous analysis of fault assumptions.

The main novelties of Project work in this task are in proposing fault tolerance templates supporting formal system development, in following a systems approach to building fault tolerance and in formulating rigorous techniques for stating fault assumptions. This task will be developed and evaluated in the context of all RODIN case studies.

T2.4 Development templates for reconfigurability, adaptability and mobility

Open system development requires rigorous support for adaptivity and mobility. To be adaptable the next generation applications should be able to dynamically react to changes in the environment, in the system behaviour and in the requirements. Reconfiguration is known to be the main means for achieving adaptivity. Mobility will allow system components to move or to be moved to achieve the system goals. Adaptivity and mobility add new dimensions to system complexity and as such require formal modelling supported by refinement as well as typical templates and guidelines helping developers to introduce these concerns systematically and without errors.

In this task we will identify general abstractions expressing adaptivity, reconfiguration and mobility in the earlier system models and investigate decomposition and mapping principles supporting stepwise development of such systems. For adaptivity we will introduce abstractions related to system/environment monitoring, (dynamic) identification of the reconfiguration units and policies, reliable and consistent changes of the system structure. Action Systems will be mainly used to demonstrate the general development method and the supporting templates. For mobility we will rely on a general system model combining both device and code mobility [Fuggetta et al 1998] (e.g. the coordination paradigm) which will be extended to incorporate fault tolerance support using exception handling [Di Marzo and Romanovsky 2003]. The typical abstractions will be related to component location, relocation and connectivity. The general method and templates will be demonstrated using several formalisms (most likely a π-calculus based one and Action Systems).

Developing abstractions and templates capturing system reconfiguration, device and code mobility and adaptivity in the context of rigorous stepwise development of fault tolerant systems is the main novelty of Project work in task T2.4. Both mobile telecoms case studies and the ambient campus will be used to evaluate the results of this task.

T2.5 Requirements evolution and traceability

Traceability plays an important role in *connecting* the *informal* requirements of a future development to the various *formal*, and gradually refined, models expressing in a rigorous manner the properties described more loosely in the former. The intent of traceability is to show that the "formal" constitutes a faithful (and probably more precise) representation of the "informal" providing assurance that the requirements are fulfilled by the formal model. Traceability is also important during maintenance of the system, or when accommodating changing requirements during the system development, as it can help to indicate the downstream impact (in terms of complexity, cost and effort) of proposed modifications of or additions to the system requirements. The experience of ClearSy and Praxis is that this form of impact analysis can in some cases be used to justify the rejection of change requests that would lead to no significant difference in behaviour.

This task will develop an approach to traceability based on adding more structure to informal requirements documents. In this approach, traceability needs to be *prepared* in the informal requirement document by isolating and *labelling* some individual short natural language statements. The requirement document is thus made of two distinct pieces of texts: the labelled one that will be part of the traceability process and the free ones that will not but are nevertheless important in order to explain the former. Traceability is then implemented in the formal model by pointing in it to the informal document by means of these labels. Once this is done at the first levels of abstraction, traceability should be pursued within more refined models until the very end of the design process reaching final code. Traceability may also be employed to link a verification proof with the items that are the subject of the proof. If either item is modified, it is then obvious which proof must also be modified. As a final outcome, one is then capable of tracing with precision the way the informal labelled requirements are eventually translated into some final implementation, as well as tracing claims of correctness to their justification. The work on traceability will provide input the tool development work in WP3 and WP4. We will also use ideas from, and were appropriate make use of, existing tools supporting

traceability in standard (non-formal) development.

Nokia's MDA, CDIS and ambient campus case studies are used to evaluate the task.

7.1.3 WP3 – Open tool kernel

Objectives. The intent of this workpackage is to develop some *basic kernel tools* implemented on a certain *platform container* that can be extended by the *plug-ins* being developed in WP4. In what follows, we shall explain what is meant by this terminology, namely "basic kernel tools", "platform container", and "plug-ins". In the course of these explanations some other terms will be introduced and thus also explained.

The role of the kernel tools is to support formal developments made in *Event-B*, which is an extension (and also a simplification) of the *B language* used in the *B Method*. This extension is specially devoted to accompany the *systems approach*, which is the basis of our proposal and which were explained in section 2. Event-B is an important continuation of B, which itself was a continuation of Z. Event-B has been influenced by the work done earlier on Action Systems by the Finnish School (Action System however remained an academic project). Event-B has been studied so far by J-R Abrial and others in France and England. Event-B is thus a synthesis between B and Action System. B has been used very successfully in the development of large real industrial software projects. Event-B, as it is now, is just a "paper" extension of B. It has proved so far to be feasible. This is the result of a number of experiments performed by using it and later simulating it on the tools of B (Atelier B). *This finalization is thus one of the initial objectives of this workpackage*.

Event-B extends the usage of B to systems that might contain software but also hardware and pieces of equipment. In that respect, one of the outcome of Event-B could be the proved definition of systems architectures and, more generally, the proved development of, so called, "system studies", which are performed before the specification and design of software. This enlargement should allow us to perform failure studies right from the beginning in a large system development. It is worth noting that a considerable number of RODIN partners already have a significant experience in using it. Moreover, Event-B is now mature enough to enter into a more active phase, namely that of building an industrial platform entirely devoted to it. In many respects Event-B is simpler, and at the same time more powerful, than B and Z, this is why we believe that the kernel development we envisage could take advantage of this but also use the ideas and techniques that have been used successfully in B.

The mechanical support of the Event-B developments alluded above was done so far by using *Atelier B* (developed by ClearSy), which is the tool associated with the *B Method*. This tool is now fully industrial: it has been used, for example, to support the formal development of the safety critical part of the embarked software developed for the new Paris automatic Metro. Using *Atelier B* to support developments made in Event-B is however *too indirect* (since the original *B Language* and corresponding tool *Atelier B* were not originally devoted to it), *too heavy* (same reasons) and *too closed* (this is due to the original closeness of *Atelier B*).

The novelty of our work in this workpackage essentially consists in transforming various ideas and attempts (developed so far as exploratory projects) about Event-B into a

coherent body (for the theory and for the tools) so that the result of our work in RODIN could be used in large industrial system engineering projects.

The kernel tools we intend to develop in this workpackage will thus quite naturally be based on those of *Atelier B* but with two more objectives in mind: *simplification* and *openness*. These two objectives are clearly interrelated. Their intent is to rationalize and give a solid industrial basis (as is already the case with *Atelier B* for "classical" B) to the development of complex systems made in Event-B. The kernel tools developed in RODIN are thus the minimal tools which have to be used to develop systems using Event-B. They are made of three categories of tools: (1) the *low level tools*, (2) the *intermediate level tools*, and (3) the *upper level tools.*

The *low level tools* are very classical. They can be encountered in any compiler: lexical analyzer, parser, type checker. These are usually collectively called the static analyzers. *Their development is not part of RODIN.* The corresponding tools of *Atelier B* will simply be slightly *adapted and simplified* in order to be able to handle the peculiarities of the language of Event-B.

The *intermediate level tools* are encountered when one supports formal developments *with proofs* (*VDM*, *B*). They are usually called *Proof Obligation Generators*. They generate, out of the formal texts submitted to them, the formal statements to be proved in order to ensure that such texts are "correct". In our case, we have two such tools: (1) the "well-formedness" Proof Obligation Generator which is devoted to generating statements for proving, as its name indicates, the *well-formedness of set theoretic expressions* used in Event-B, and (2) the "system property" Proof Obligation Generator which handles all Event-B system property statements, namely safety, liveness, refinement, generic instantiation, and decomposition. The corresponding tools of *Atelier B* only partly support such properties. Although their external structure might be re-used, a more substantial effort (than the one used for the low level tool adaptation) will be necessary to adapt them to our needs. It might be the case that *a significant re-development of the second one will be necessary in RODIN.*

The *upper level tools* correspond to what is usually called the *Provers*. Such tools are able to handle either automatically or interactively the statements generated by the Proof Obligation Generators by discharging them. The provers of *Atelier B* (there are two of them) will be *largely reused in RODIN*. The development of new provers is thus not one of our objectives in RODIN. We shall only *simplify and adapt certain rules and procedures of these provers and slightly extend them.*

The most significant part of the RODIN effort concerning these tools will thus be devoted to the *organization of their openness* because the ultimate aim of all project efforts in WP3 is to deliver this platform openness. This is where the consortium intends to make considerable advances in the state of the art. Many analysis tools, such as model checkers and theorem provers, have been developed. However a major drawback of these tools is that they tend to be closed and difficult for other interested parties to extend making it difficult for the work of a larger research community to be combined. The RODIN open tools kernel will greatly extend the state of the art in formal method tools allowing other parties to integrate their tools as plug-ins to support RODIN methods.

Such an objective means that extra tools, collectively called the "plug-ins", will be easily added to the basic tools (some of these plug-ins are described in WP4, but the list given

there is by no means limited: it is indeed open). Openness is very important in that it will allow future industrial users to *customize and adapt* the basic tools to their particular needs. For example, a car manufacturer using Event-B to study the overall design of a car information system might be willing to plug some special tools able to help defining the corresponding documentation and maintenance package. Likewise, a rocket manufacturer using Event-B might be willing to plug a special tool for analyzing and developing the failure detection part of its design.

This openness effort will be materialized by three distinct realizations: (1) the definition and implementation of a *Connection Language*, (2) the design and implementation of a *Project Manager*, and (3) the construction of the *Open Platform* itself.

The *Connection Language* is the "universal" means by which the syntactic structure of Event-B texts can be exported to the plug-in tools. It will be based on XML and the generation of this syntactic structure will be performed by the low level basic tools (after type checking).

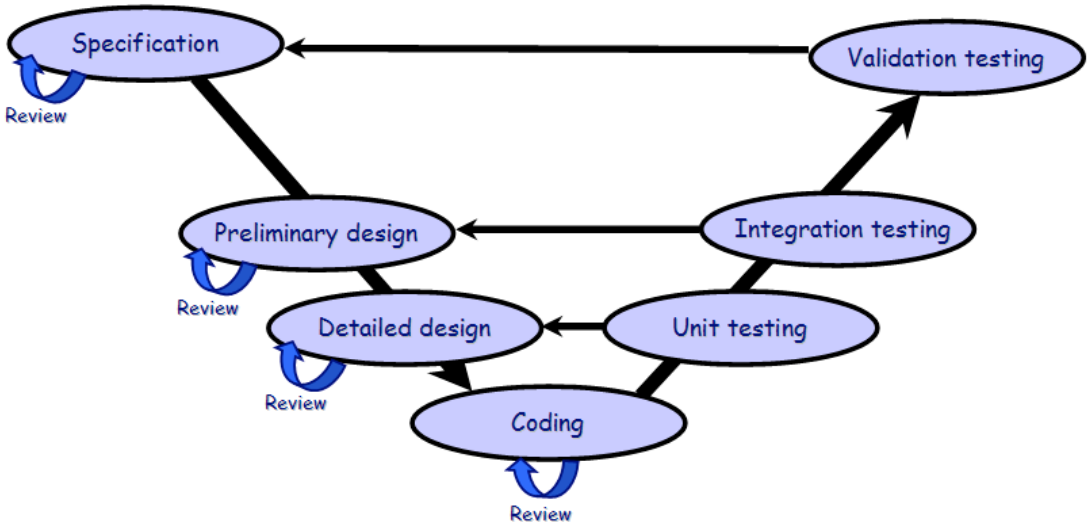
The *Project Manager* is the tool, which is able to synchronize, record, reset and possibly resume the various executions of either the basic tools or the plug-ins devoted to a specific industrial project. Without such a tool (which is a kind of "Make" tool), the handling of a large projects (involving many users) would very quickly become highly chaotic. Such a tool exists in *Atelier B*, but it is organized around a fixed number of pre-defined tools and thus not able to handle the dynamics of our plug-ins.

The *Open Platform* is the ultimate software structure implemented on the user Operating System (Linux, Windows) and supporting the basic tools and some of the plug-ins. *It is not our intention however to develop a specific new platform in RODIN*. It will be part of an initial study to determine which existing platform container we could use (for example, Eclipse might be a candidate).

To summarize at this point, here are our various objectives to be achieved to ensure the openness of the platform.

1. Finalization of the Event-B *Language*.
2. Adaptation of the *Low Level Basic Tools: Static Analyzers*.
3. Adaptation of the *Intermediate Level Basic Tools: Proof Obligation Generators*.
4. Adaptation of the *Upper Level Basic Tools: Provers.*
5. Design and implementation of the *Connection Language*.
6. Design and implementation of the *Project Manager*.
7. Design and Implementation of the *Open Platform*.

Software development is based on a typical V-cycle organisation. Specification and design documents are internally reviewed and assessed. Those documents would include formal specification if required, and would drive the writing of the test case specification (unit, integration and validation testing). Eclipse embeds testing facilities based on JUnit library, which will allow us to seamlessly develop V&V activities.



Source code will be reviewed, according to different criteria (readability, robustness, comment ratio, cyclomatic complexity etc.) in order to ease any further maintenance and evolution of the software. In addition, static analysis tools are to be used with Java and theory language, in order to increase confidence. Eclipse provides some useful capabilities to facilitate Java code audit (e.g., dead code, cross reference). Moreover ClearSy has internally developed tools for theory language source code which will be used during the project.
Interface specification takes into account the platform architecture, the connection language and the preliminary specification of the intended plug-ins. This specification requires a consensus within the consortium. This specification is likely to evolve during the lifetime of the project. Alpha and beta tests will be organized with partners, IIG members and external volunteers.

Description of Work

The workpackage will consist of different tasks contributing to common deliverables as follows:

Task 3.1 Finalization of Event-B Language
The finalization effort of the Event-B Language will focus on some important aspects of this language concerning various possibilities of *extensions*, namely that the *mathematical language* and that of the *modelling language*. In "classical" B, the mathematical language cannot be dynamically extended as it is the case in some other formal languages (for example in Z). This proves to be an unacceptable limitation in some cases, in particular when one deals with complex data structures. The modelling language will receive some possibilities of extension so that the user will be able to define *generic models* and later instantiate them (adaptive re-use).

Task 3.2 Adaptation of Low Level Basic Tools: Static Analyzers
This task first concerns the final definition of the Event-B Language *lexical and syntactic structures*. Once this will be achieved, the corresponding tools will be adapted

accordingly. This task is also connected with **Task 3.5** where the Connection Language is defined since the generation of such a connecting structure is a direct external representation of the parsing of an Event-B formal text. As a consequence, the parsing tool has to be extended to handle such a generation.

Task 3.3 Adaptation of Intermediate Level Basic Tools: Proof Obligation Generators
As the Event-B Language is able to express *more properties* than the classical B Language, the "system property" Proof Obligation Generator has thus to be significantly enhanced. Besides the traditional correctness properties (invariance and refinement) already existing in classical B, Event-B deals with the following new properties: (1) the variant properties to be proved when new events are introduced while refining, (2) the deadlockfreeness property expressing that a refinement does not deadlock more often than its abstraction, (3) the necessary properties to be proved while instantiating a generic model or a new generic mathematical operator, and (4) the reachability conditions that one is able to express in Event-B. All such properties need to be treated by the new "system property" Proof Obligation Generator: they express the semantics of the Event-B Language. The "well-formedness" Proof Obligation Generator has also to be extended as one is now able to construct new *partial* mathematical operators in Event-B. This partiality has to be taken care by the "well-formedness" Proof Obligation Generator when encountering such a partial operator in a formal text.

Task 3.4 Adaptation of Upper Level Basic Tools: Provers
The Prover of *Atelier B* will be *simplified* as some mathematical constructs will completely disappear (for example sequences and trees). In fact, they will be replaced by similar constructs obtained by the mathematical extension mechanisms. In practice, the provers will only handle the following theories: (1) Propositional Calculus, (2) First Order Predicate Calculus with Equality and Pairs, (3) Elementary Set Theory (basic set operators, binary relations and functions), and (4) Elementary Arithmetic. All other mathematical structures will now be constructed externally by means of the extension mechanism. However, the Provers might be slightly extended to handle Arithmetic in a more efficient way although some arithmetic operators (for example, modulo) will also be defined by extension.

Task 3.5 Design and Implementation of the Connection language
Since XML has become a "de facto" syntactic standard and is now supported by a large number of tools and applications, RODIN tools will use XML format for its connection language. The first important sub-task to be performed here is the design of an XML representation for the artifacts to be exchanged (formal models, proof obligations, etc). This XML representation is important for the design of the various plug-ins in WP4. Once this XML representation is designed, the second subtask will be to extend the parser (**Task 3.2**) in order to generate the appropriate XML after analyzing a formal text.

Task 3.6 Design and Implementation of the Project Manager
As explained above, this Project Manager is a kind of "Make" tool. The presence of the various plug-ins that may be added to one or several projects clearly imposes that this tool can be customized by corresponding files (as the "Make" tool working under UNIX is). As

a consequence, the first sub-task will consist of investigating the possibility to use an existing tool. Once this is done, the second sub-task will be to adapt our choice to the specific structure of our platform. In this task, the connection with the plug-in tasks of WP4 must be clearly identified so that we can be certain that the various plug-ins are indeed well integrated with the Project Manager.

Task 3.7 Design and Implementation of the Open Platform
The first sub-task is again to investigate the "platform container" that we could chose (again we have Eclipse in mind the definitive choice is not yet made). Once, this is done, our second sub-task will be to port our various basic tools and integrate them in the platform. We will make use of appropriate XML technology to support this (e.g., the Eclipse Modeling Framework).

7.1.4 WP4 – Modelling and verification plug-ins

Objectives. This workpackage will develop a range of tools to support the application of the RODIN methodology being developed in WP2. These tools will integrate with open tools kernel being developed in WP3. Early versions of the tools will be tested through application to the case studies of WP1, leading to improvements in functionality and design in the tools.

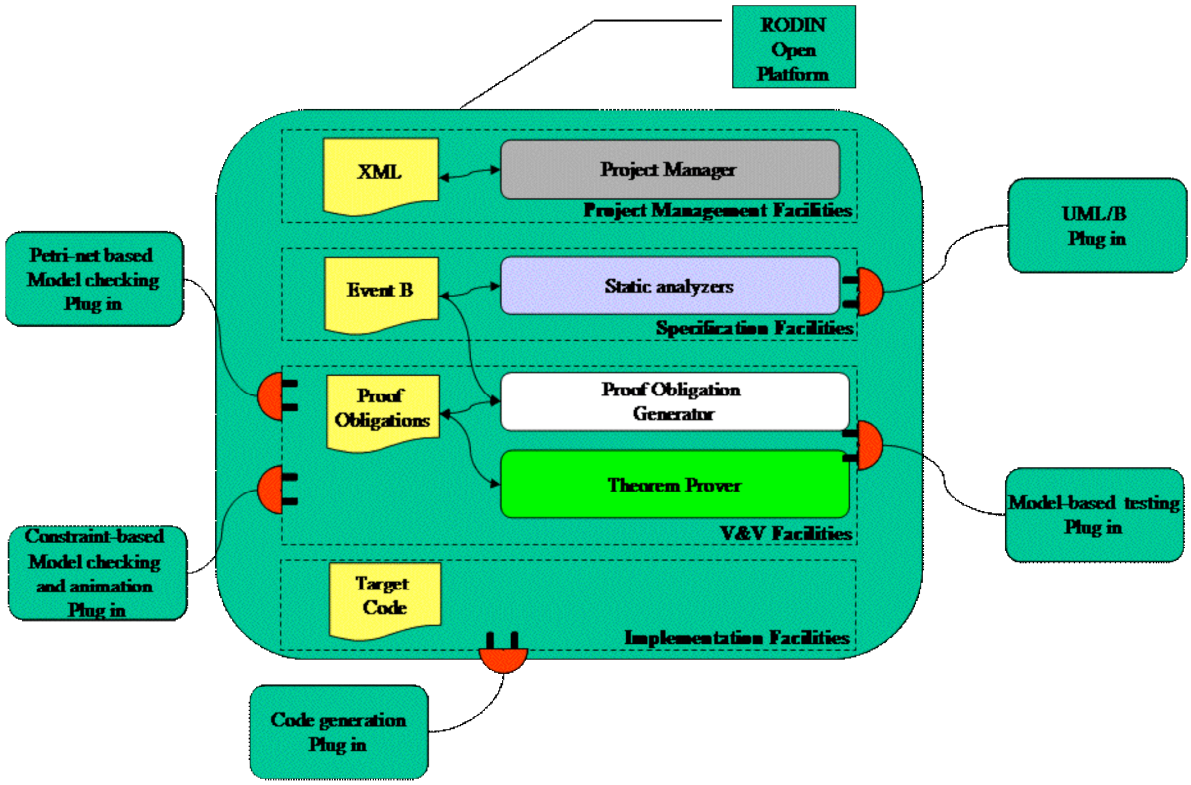
The more specific objectives are to build a range of plug-in tools providing the following key functionalities:
1. Linking UML and B
2. Petri net based model checking
3. Constraint-based model checking and animation
4. Model-based testing
5. Code Generation

We will consider the development of other plug-in tools during the lifetime of RODIN if the need arises from the case studies or the methodology research.

The model checking tools should be seen as complementary to the deductive proof technology being provided by the open tools kernel. For example, model checking could be used to find errors in abstractions of specifications before embarking on full deductive proof or could be used to attempt to falsify proof obligations generated by the kernel. Our philosophy is the more tools the systems engineer has at their disposal, the more effective they will be. Because model checking is technically challenging and different forms of model checking are suited to different forms of models, we plan to develop model checking tools based on complementary approaches based on constraint logic programming and based on Petri net tools.

The relationship between the tools kernel (WP3) and the plug-ins is illustrated by the following diagram:

Description of Work



The workpackage will consist of several concurrent tasks each contributing to common deliverables as follows:

Task 4.1 Linking UML and B

UML is a semi-formal design notation that enjoys widespread use in industry as a language for modelling computer systems. UML is based on the object-oriented paradigm and makes strong use of graphical notation including class diagrams, state charts, use-case diagrams, and message sequence charts. UML is perceived as being more accessible to engineers than formal methods such as B probably because it requires less mathematical training, the notation is quite. Unfortunately, specifications and designs written in UML are not amenable to the rigorous correctness proofs that are supported by methods such as B, and thus UML on its own is not adequate for the development of correct-by-construction computer systems, which is the major focus of RODIN. Southampton have already developed a tool (U2B) that allows UML class diagrams to be annotated with B descriptions and then converted into full B descriptions, which are amenable to formal analysis. Thus systems can be specified using a combination of UML and B and then formally analysed in the manner of a B specification. Currently the U2B tool supports UML class diagrams and state diagrams. Such a tool can play a major role in bridging the gap between the techniques used by design engineers and formal specifications in B.

This task will focus on ways of providing better representations of the following key concepts in UML: fault tolerance, component reuse and MDA. It will define appropriate

translations from these UML representations into the underlying formalism and extend the existing U2B tool to support these conversions. The tool will be integrated with the open tools kernel. We will investigate the use of the OMG Meta-Object Facility (MOF) to represent mappings between models. The task will work closely with most of the industrial case studies. In particular the support for fault modelling and component reuse will be developed in collaboration with the embedded engine control case study while the support for MDA-style development will be developed in collaboration with the mobile telecoms case studies and the air traffic control case study. The main novelty here will be to provide a formal basis for the MDA approach and to extend the existing UML-B tools to support component reuse and modelling of faults and fault tolerance.

Task 4.2 Petri-net based model checking

Model checking of concurrent systems is intrinsically hard, and exhibits a trade-off between the compactness of the representation of the system and resources it takes to verify behavioural properties. For example, the deadlock detection problem is PSPACE-complete for a compact (bounded) Petri net representation, but polynomial for transition system representation. However, the latter is itself often exponentially larger, and soon becomes too large to be stored in the main memory, which makes the algorithm impractical. The method we will exploit in this task in the development of the model checking plug-in, based on a complete prefixes of net unfolding, offers a compromise: its size is in between those of the original specification and the corresponding transition system, while typical verification problems are NP-complete. The existing experimental results demonstrated that unfoldings can be very efficient for several crucial verification problems and application areas; in particular, telecommunication systems. The aim of our work here will be to extend the existing model-checking technique based on unfoldings ([Khomenko et al 2002]) to systems with mobility, where the communication links can change, and computing agents migrate, as the system evolves. Our work will take as the starting point the π-calculus [Milner et al 1992] (a standard process algebra used to describe systems with mobility), develop for it a translation into a suitable class of high level Petri nets, and implement a plug-in for model-checking specifications expressed in the latter. In this task we will extend the existing model-checking technique based on net unfoldings to systems with mobility, the importance of this task is in complementarity of model checking to the deductive proof techniques supported by the majority of the open tools plug-ins. The novelty of this work is in focusing on a high degree of concurrency typical for systems built out of mobile processes. Net unfolding technique will be the first model-checking tool aimed specifically at the verification of mobility systems using fully automated model-checking aimed at dealing with state space explosion. The model-checking plug-in for mobility systems will be developed and evaluated in the context of the mobile telecoms systems and the Ambient Campus case studies. The usability of the plug-in will be also assessed in an industrial scale concurrency specification of the air traffic control system.

Task 4.3 Constraint-based model checking and animation for B

ProB is a prototype animator model checker for B developed at Southampton. It is based on a prolog implementation of an interpreter for B and uses sophisticated techniques such

as constraints, co-routining and program specialisation to achieve optimizations in model checking. The tool can be used to animate B specifications at different levels of abstraction which helps systems engineers to validate that requirements and design choices have been properly modelled. The model checking mode can be used to find possible ways in which a model may become inconsistent, i.e., have a state that violates desired invariant properties. The tools does not currently support checking of refinement between models.

In this task, the ProB tool will be extended to deal with automated checking of refinement between models and will be integrated with the open tools kernel. The tool will also be tailored to working more closely with the deductive prover in the open tool kernel so that individual proof obligations, and not just whole models, may be checked for possible violations. The constraint-based animation and model checking plug-in will be developed and evaluated in collaboration with all of the industrial case studies since they all will involve development through refinement. The novelty here will be a very tight integration of constraint-based animation and model checking with the B development environment to provide richer development functionality.

Task 4.4 Model-based testing

Model-based testing involves the automated generation of test cases from a formal model of a system that can be used to test the implementation of that model. As well as being used to generate test cases, the mode can also act as the oracle, determining what the outcome of test cases should be in the system under test. As with model checking, this provides further analysis tools in the armoury of the systems engineer that complement the use of deductive proof.

This task will develop a plug-in tool for model-based testing. The tool will be based on an interpreter for the underlying formalism, such as the B interpreter used in the ProB tool. Notions of test coverage will be defined based on standard techniques for partition analysis in model-oriented formal specifications. This is a form of black box testing since the test coverage is based on the structure of the specification rather than on the implementation structure Test case generation algorithms based on path traversal will be developed with the objective of achieving high degrees of coverage in testing. A further stage will be linking the test case generation with particular target implementation languages.

This level of tight integration of model based testing, that exploits formal specifications, into a development environment is novel

Task 4.5 Code generation

Event-B can be used to model sequential and distributed algorithms. After some refinements, one eventually obtains a completely deterministic model which is liable to translation into a classical programming language such as C, C++ or Ada. That translation consists of first translating every event of the model in the target language and then combining those translations using syntactic rules to produce sequence, conditional and

loop constructs. This technique allows developing programs correct by construction as it was the case with classical B. However, the use of event-B allows one to prove more easily the correctness of the program (proofs are much smaller than in classical B) as it has been shown in case studies. It also allows developers to take advantage of the other tools of the open platform.

The main novelty is in the production of correct-by-construction source code from event B models, allowing to obtain executable implementation of event-based software (sub)systems.

7.1.5 WP5 – Dissemination and exploitation

Objectives. The goal of this workpackage is to manage the diffusion of the information and to make precise the exploitation plan of the outcomes of the project, mainly the open platform. The first part relies upon three kinds of actions: rapid transmission of the information between the partners, diffusion of the results outside the project, towards the academic community and towards the industrial world, links with the Industrial Interest Group. On the second part, the workpackage determines the strategy of maintenance and exploitation of the RODIN open platform for rigorous development of complex systems, during and after the project.

Organisation. An industrial partner, ClearSy, with a strong experience in dissemination and exploitation is leading this workpackage. ETH, as an academic partner, is assisting ClearSy in all work related to academia. ClearSy is responsible for the practical organisation of the dissemination and exploitation of RODIN's results.

Description of work

Task 5.1 Transmission of information between the partners
A **web site** will be opened and maintained by the RODIN consortium. It will provide a centralized mailing list and will allow rapid diffusion of information between the partners. It will contain a public part and a private part for draft documents. The open-source tools of the RODIN platform will be downloadable from this site. Open-source tools will probably be released using Common Public Licence policy, as this policy provides a great liberty concerning the development and the selling of further plug-ins / extensions. The platform remains open-source and freely downloadable, while plug-ins can be either free of charge or not. GPL licence requires that any product including GPL parts becomes GPL, meaning providing the source code of any plug-ins. Moreover, the Eclipse framework comes along with the CPL policy.

Task 5.2 Diffusion of the Results

A strong publication policy will be promoted. All major project results will be presented and discussed in the main conferences and symposia in the related areas and submitted for publication to the major journals. Information will be provided to the academic community and from this community to the RODIN partners (event announcements). The partners will maintain links with Special Interest Groups (e.g. B User Group, Z User Group, FME, etc.). The publication of scientific papers will be under the responsibility of the leaders of the workpackages. The industrial partners will present their work at

industry-sector focused conferences such as BCS interest groups and the Safety Critical Club.

The RODIN consortium will manage a steering committee to organize **International workshops** (in association with major International Conferences or Symposia, such as FME, ETAPS, DSN, ZB held in Europe). These workshops are intended to widely disseminate RODIN advances, to discuss project results with the wider research communities and to be forums to intensively exchange with researchers working in the similar fields. Two International workshops are definitely planned: the first workshop will be held by month 18 (a likely venue is FM 2005, the 13th International Symposium of Formal methods Europe) where the partners present the work already done and open up the perspectives for the rest of the project. The Industrial Interest Group will be specifically invited to attend the meeting, and also any interested people. A second International workshop will be organized by month 30 of the project as an integral event with large publicity, call for paper and call for participation. It will contain one or several session(s) on the RODIN results, and demonstrations of the open platform and tools.

Tutorials should be designed and carried out with the partners in order to present the most advanced state of the art on the RODIN topics. In the same flavour, the tools and platforms developed by the partners will be demonstrated at the scientific workshops and conferences above mentioned.

The outcomes of the RODIN project will have impact on software development and on construction of fault tolerant systems. The academic partners plan to write **educational materials** on these topics for university courses in software engineering. They will exploit the results of RODIN in their undergraduate and postgraduate teaching. The B Method is already used at some level courses, so a smooth integration is possible. The partners will make teaching material based on RODIN methods and tools available for other academic institutions and for the formation of industrial people. After suitable simplification, the cases studies developed in the project have vocation to become examples of developments for a better understanding of the B method.

Although the main role of the case studies developed in WP1 is to act as research drivers, they will play an important role as demonstrators and as aids to technology transfer of RODIN methods. They will be used to strengthen the impact of the major results of RODIN: its methodology and tools platform.

Task 5.3 Links with the Industrial Interest Group

Contacts are numerous with companies that are interested in experiences about the integration of formal methods, and specially the B method, in the software lifecycle. These companies or industrial teams will be associated to the project without commitment, as observers. They receive information about advancement and results, and are primarily invited to the workshops organised by the RODIN project. They can be more closely associated, if they express the need for that, in a way which will be determined and accepted by the partners. The initial list of the Industrial Interest Group is provided in Section 5. All the companies have duly expressed their interest in the RODIN objectives and results.

Task 5.4 Specific contributions to the exploitation plan

The general dissemination aspects applicable to all the partners, are listed in task 5.2. In tasks 5.4 and 5.5, the specific contributions of the partners are detailed.

University of Newcastle upon Tyne

The Ambient Campus case study has been chosen both for its technical relevance and for their ability to facilitate exploitation of the project results. It aims to apply and evaluate the proposed method of rigorous stepwise development of fault tolerant mobile systems and the platform developed within the project in the typical pervasive mobile environments, which are now becoming available in a variety of organisations.

Newcastle will make full use of its many industrial contacts in order to ensure that the results of RODIN are effectively and promptly exploited. These include the British Aerospace-funded Dependable Computing Systems Centre, the Centre for Software Reliability (CSR), which has for years run very extensive industrial technology transfer programmes. In particular, CSR runs two industrial community clubs, in software reliability and in safety-critical systems, which disseminate results and problems to 2,500 members via about 10 events each year.

Aabo Akademi University

Aabo will exploit RODIN results in its future projects via our centre of excellence, CREST and its industrial contacts. Especially the results from our work in WP2 on the methodological issues will be the focus here. Aabo will work closely with Nokia in integrating the results of RODIN into their methodology for protocol engineering. Our work on introducing fault tolerance in system development will contribute to the VTEC case study. In our previous work we have successfully used the combination of UML and B within stepwise systems development. This work will contribute to the MDA approach within the Nokia case studies.

Nokia

Work is planned to disseminate and utilise the techniques in complex and large projects. From Nokia Research, dissemination is made via consulting, coaching, internal conference and teaching with the relevant business units.

Praxis Critical Systems

Praxis Critical Systems intends to use the notations and tools from the RODIN project directly in future industrial-scale software development projects for critical systems, in aviation, transport, security and finance. We already have an extensive track record in publishing our experiences of the use of formal methods and tools on industrial projects.

VT Engine Controls

The project will produce the basis of a formally specified and designed generic package for configuring failure management systems. Subject to company business strategies this may be developed into a viable product and used by VTEC to produce future safety critical products efficiently. It could also (or alternatively) be sold as a package for other companies to use in developing their software. Experience gained on the project will contribute in the methods used to develop and support future products. Of particular interest in the avionics industry is the safe use of object-oriented methods.

ETH

ETH intends to use the RODIN platform as a basic toolbench for its work on trusted components and especially the systematic development of contract-equipped classes whose implementation is proved correct with respect to the contracts. ETH has a strong connection to local industry (both the financial industry and engineering companies); its successful program of short courses for professionals will be used to promote the results of the research described here.
ETH will promote the open source availability of the RODIN platform. This will be enhanced and marketed by ClearSy with whom there are very strong links both because of Abrial's former link with them and because Laurent Voisin is moving to ETH. The dissemination of RODIN plug-ins is more open but ETH will install all of them to show to interested organisations. Suitable demonstration examples will also be available at ETH for such plug-ins.

In addition to these industrial applications, the results are expected to influence the software curriculum, which the ETH project members are progressively reshaping to include a stronger dose of formal concerns.

University of Southampton

The University of Southampton will work closely with VTEC in helping them to integrate the results of RODIN into their product development. Southampton will also exploit its strong contacts with other companies to disseminate the results of RODIN, especially the platform and plug-ins. Our work on combining UML with B and applying this combination to stepwise development and the MDA approach will contribute to providing a more rigorous basis for the MDA approach and, with support from Nokia, we will contribute to OMG standards in this area.

Southampton will exploit RODIN results in future projects funded by industry, UK bodies and the EU. As part of WP4, Southampton will produce two plug-ins for the RODIN platform: a model checker for B (based on our existing ProB tool) and a tool for linking B and UML (based on our existing U2B tool). These tools will be made available for teaching and research use. We will also investigate the setting up of a spin-off to produce commercial version of these tools. Southampton has a very strong record of setting up successful spin-off companies and the University provides excellent support for this. The combination of papers at workshops and journals and the availability of plug-in tools from Southampton will make an impact that will contribute towards its reputation as a world class teaching and research institution.

Task 5.5 Strategic plan for the platform maintenance

The objective of the RODIN consortium is to market the resulting platform as a systems engineering product. To be in line with this objective and with the potential market, the consortium will be responsible for its dissemination and will envisage creating during the lifetime of RODIN a dedicated subsidiary or an association. The rationale for setting up a subsidiary would be driven by the positive feedback gained from IIG and other parties, and by our capability to demonstrate RODIN's technology and method added value. The subsidiary or the association will be in charge (from its creation or from the end of the project) of:

- Developing the platform: The platform and most plug-ins will be released as open source tools, and managed via sourceforge.net facilities (distributed development, CVS support, mailing list, news). The proof tool plug-in, derived from existing Atelier B tool (representing tens of men year work), will be licensed and sold as a product. The consortium will also take part in the technical discussions and decisions while extending / improving the platform.
- Promoting and disseminating the platform: The dissemination of that platform will use Atelier B installed base (about 650 licences worldwide) as well as the industrialists who still have experimented B on their own. Dissemination will also be based on Steria's[1] and Teamlog's commercial capabilities[2]. Based on previous projects and experiments, many customers are still interested, namely in the domains of:
    - safety cases: railways signalling devices design/validation/certification, interlocks in mammography scanners, FDIR strategy (micro satellites),
    - embedded diagnosis: generation of reliable diagnosis data,
    - specification expertise: verification of statement of work, specification / design validation.
    - system development: SoC, numerical TV, etc.
    - reverse engineering: nuclear control plant,

  Those activities, which were previously conducted by hand, will benefit from being automated and concentrated in a single, industrial strength environment. Moreover, some high tech Steria's and Teamlog's subsidiaries may also by used as preferred dissemination vectors, namely Imelios (networked telecoms) for Steria, Firstel (telecoms) and @sky (satellite TV) for Teamlog. The consortium members will use the platform as basic tool inside industrial projects.
- Building dedicated plug-ins: Based on industrial and academic feedbacks, specific plug-ins will be developed by skilled team. These developments may include academic and / or industrial partners, and cover very different aspects (Human Machine Interface, code generator, process wizard, etc.).
- Ensuring consulting support: The consortium will provide technical consultancy for evaluations and operational deployment.
- Organizing training sessions, on demand, in Europe as well as in the US: Training sessions will be organised, to address specific needs: method discovery for project managers, in-depth navigation for practitioners.

All these activities will be funded by:

- incomes provided by plug-ins development and sale, training sessions and consultancy activities,
- National grants and funding.

[1] ClearSy Systems Engineering is a joint subsidiary of Steria and Teamlog. That company has been set up to address more accurately the formal methods and systems engineering domains.

[2] Steria is present in France, UK, Spain, Portugal, Germany, Belgium, Denmark, Norway, Sweden, Switzerland, Singapor, Saudi Arabia - Teamlog is present in France, Spain, Portugal, Switzerland, Hungary, Canada.

Some partners are still structured for these activities and have more than 10 year experience in that domain, with skilled teams (sales, training, consultancy, developers, maintenance).

Task 5.6 Dissemination and exploitation organisation

The objective of this task is to organize dissemination and exploitation activities during the lifetime of the project and to ensure / to ease, as far as possible, final adoption of the platform by both academics and industrialists.

Initial Dissemination Report (D5.1) will be prepared to outline an initial financial plan for use and dissemination of knowledge, to raise public participation awareness and to report Project initial work on dissemination and exploitation.

Successive releases of the Dissemination and Exploitation reports (D5.4A, D5.4B, D5.4C) will be provided. This document will contain the following sections:

- dissemination and exploitation plan: definition of a dissemination and exploitation strategy, action items undertaken or to undertake in order to achieve our objective. For example:
    - specification/development of new plug-ins required to address (specific) needs
    - build synergy with other projects, which could complement / strengthen our activities.
- dissemination and exploitation financial plan: elaboration of a "business plan" for assessing/validating further commercial exploitation, including creating a dedicated subsidiary.
- "Public Awareness" report: involvement of actors beyond the research community, measurement of RODIN website foreign hits, IIG feedback.

This document will be initiated during the first year, released at M12, M24 and M36.

7.1.6 WP6 – Project Management

The RODIN Executive Board, chaired by the Project Coordinator will guide the overall conduct of the project, setting overall technical goals, coordinating and reviewing technical progress, revising the project plan as necessary, ensuring the timeliness and quality of the deliverables, and developing an exploitation plan. It will be assisted by the Administrator, who will act as Secretary to the EB. Membership of the Executive Board will consist of one representative of each Partner, with each partner also identifying an alternate, who may attend meetings for information only, or as a full replacement for the main contact. Other individuals may be co-opted as appropriate.

The Executive Board will meet at least twice a year, normally in connection with Project internal workshops, and will also have regular meetings with the Advisory Board. Additional meetings will be called as and when necessary. Detailed responsibility for the different workpackages has been divided among the Partners - who nominate particular individuals to have detailed responsibility for the completion of the work and the timely production of the deliverables.

Project Co-ordinator will delegate all administrative responsibilities to the Administrator.

The project infrastructure will use the Internet, and will consist of an internal and public web server, and an internal server for documents, code and documentation, version control and archived mailing lists.

7.1.7 WP7 – Project Review and Assessment

Objectives

The purpose of this Workpackage is to ensure that the intermediate project results and deliverables are correctly aligned to the overall objectives of the RODIN project and that there is no drift away from these objectives during the lifetime of the project.

Description of Work

Task 7.1 Establish Project Metrics

The first task of this Workpackage is to establish the metrics to be employed in the assessment of the project's success in achieving its objectives and to record these metrics in the WP7 deliverable "Procedures for Technical Review and Assessment". These metrics will be invoked in each annual Assessment Report in order to give a clear assessment of the current state of the project. The metrics will include cost-effectiveness parameters identified specifically for each case study defined in WP1, so that the case study improvements effected by RODIN tools and techniques can be assessed on a sensible basis.

Task 7.2 Review Current Technical State of Project

The evaluation of the intermediate project results and deliverables will be performed internally as a process of circular internal review and assessment. This circular review involves interaction between the academic and industrial partners in the following manner:

- the industrial partners will assess the research results and prototype tools in terms of their usability and their applicability to industrial-scale problems (and in particular the case studies from WP1);
- the academic partners will review the above assessment in order to improve the guidance for the use of the results and tools, or to modify the results and tools, as appropriate in each case.

The circular review also involves interaction between the participants from different (but related) Workpackages:

- the intermediate results from WP1 on the case studies will be assessed and reviewed from the point of view of methodology development by the participants in WP2;
- the intermediate results from WP2, WP3 and WP4 on the methodology, tools and platform will be reviewed and assessed from the point of view of the case study developments by the participants in WP1.

Internal review and assessment sessions will be held annually during the Project internal workshops in order to obtain information for the Annual Assessment Reports (T7.3). In

addition to the input provided via the review sessions, the WP7 leader will collect information from each WP leader concerning the use of the results of each workpackage by other workpackages. The WP7 Leader will also collate any review comments from the IIG members for possible incorporation into the WP7 assessment reports.

Task 7.3 Prepare Annual Assessment Reports

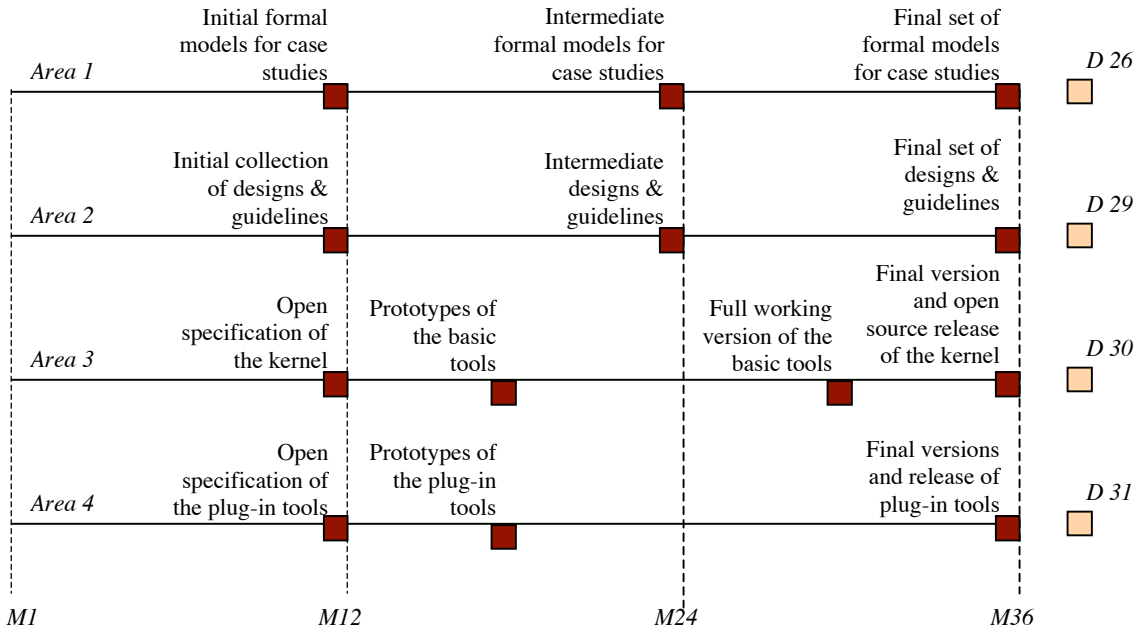
The assessment results (including the degree of divergence from the established project success metrics) will be documented and fed back to the project management via annual Assessment Reports to help in guiding the project and to the partners whose work was reviewed and assessed. Although the Assessment Reports are mainly for internal use, they will be given the status of formal project deliverables so that external bodies can also obtain an authoritative view of the progress of RODIN. Each Assessment Report will include a roadmap for expected results at the beginning of each reporting period, a review of what has been achieved by each WP at the end of the reporting period (including any perceived improvements in the cost-effectiveness of the case studies induced by the RODIN platform and methodologies), any synergies between RODIN and other projects and feedback between WPs and lessons learned

7.1.8 Roadmap of the Major Project Results

The major project results will be achieved in the following four areas:

1. Models, architectures, proofs and components (re-usable development templates) produced by the case studies
2. Design abstractions for fault tolerance, guidelines for model mapping, architectural design and model decomposition
3. Open tool kernel that supports extensibility of the underlying formalism and integration of tool plug-ins
4. Plug-in tools for model constructions, simulation, checking, verification, testing and code generation.

The split of the major project results by the areas, the time of delivering the results per area and the final deliverables reporting them are shown on the diagram below.



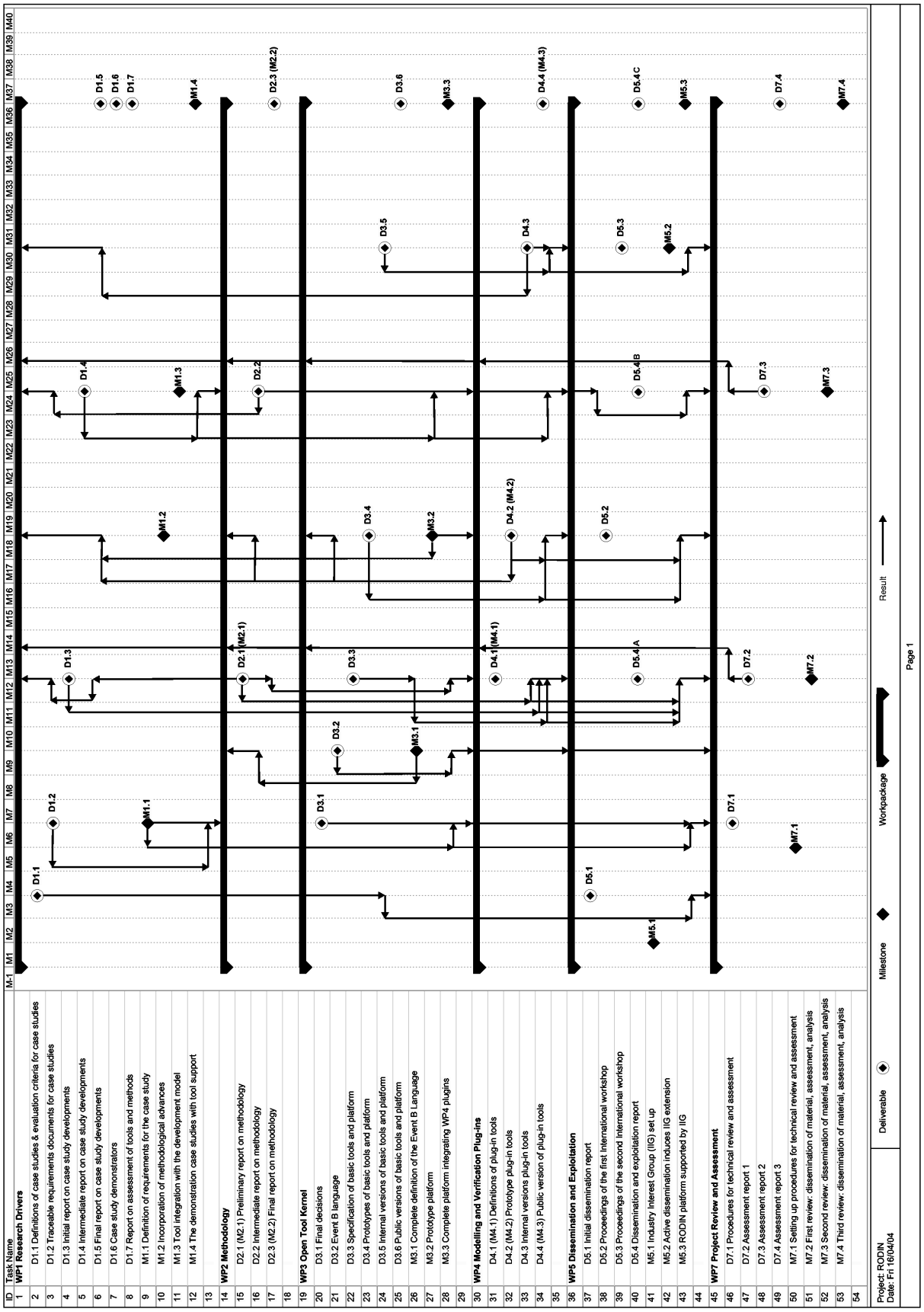
By the end of the project the RODIN consortium will produce an open environment for developing complex modern systems. We will release an open source platform consisting of the kernel and a set of plug-in tools. This platform will be accompanied by a set of guidelines which include a number of design abstractions to help the developers to rigorously build such systems. Models, architectures, proofs and components, produced during development of the case studies in the course of the project, will be released to demonstrate the best practice in using the RODIN environment. RODIN will lead the way in providing the next generation design technology to support the development of dependable complex software systems and services.

Milestones list (full duration of project)

Milestone No	Milestone title	Milestone date
M 1	M5.1 Industrial Interest Group (IIG) set up	1m
M 2	M7.1 Setting up procedures for Technical Review and Assessment	5m
M 3	M1.1 Definition of requirements for the case study	6m
M 4	M3.1 Complete definition of the Event B language	9m
M 5	M2.1 Preliminary report on methodology	12m
M 6	M4.1 Definitions of plug-in tools	12m
M 7	M7.2 First review meeting: dissemination of material, assessment, analysis	12m
M 8	M1.2 Incorporation of methodological advances	18m
M 9	M3.2 Prototype platform	18m
M 10	M4.2 Prototype plug-in tools	18m
M 11	M7.3 Second review meeting: dissemination of material, assessment, analysis	24m
M 12	M1.3 Tool integration with the development model	24m
M 13	M5.2 Active dissemination induces IIG extension	30m
M14	M7.3 Third review meeting: dissemination of material, assessment, analysis	36m
M 15	M1.4 The demonstration case studies with tool support	36m
M 16	M2.2 Final report on methodology	36m
M 17	M3.3 Complete platform integrating WP4 plug-ins	36m
M 18	M4.3 Public versions of plug-in tools	36m
M 19	M5.3 RODIN platform supported by IIG	36m

7.2. - 7.3. Planning and timetable. Graphical presentation of workpackages

The workpackages of RODIN will evolve concurrently from the beginning till the end of the project. The general timetable and interdependencies between the workpackages are shown below. WP6 is now shown as it simply collects information about results and current state of the work from to all other WPs.


ID
Task Name
M-1 M1 M2 M3 M4 M5 M6 M7 M8 M9 M10 M11 M12 M13 M14 M15 M16 M17 M18 M19 M20 M21 M22 M23 M24 M25 M26 M27 M28 M29 M30 M31 M32 M33 M34 M35 M36 M37 M38 M39 M40
1 WP1 Research Drivers
2 D1.1 Definitions of case studies & evaluation criteria for case studies
3 D1.2 Traceable requirements documents for case studies
4 D1.3 Initial report on case study developments
5 D1.4 Intermediate report on case study developments
6 D1.5 Final report on case study developments
7 D1.6 Case study demonstrators
8 D1.7 Report on assessment of tools and methods
9 M1.1 Definition of requirements for the case study
10 M1.2 Incorporation of methodological advances
11 M1.3 Tool integration with the development model
12 M1.4 The demonstration case studies with tool support
13
14 WP2 Methodology
15 D2.1 (M2.1) Preliminary report on methodology
16 D2.2 Intermediate report on methodology
17 D2.3 (M2.2) Final report on methodology
18
19 WP3 Open Tool Kernel
20 D3.1 Final decisions
21 D3.2 Event B language
22 D3.3 Specification of basic tools and platform
23 D3.4 Prototypes of basic tools and platform
24 D3.5 Internal versions of basic tools and platform
25 D3.6 Public versions of basic tools and platform
26 M3.1 Complete definition of the Event B Language
27 M3.2 Prototype platform
28 M3.3 Complete platform integrating WP4 plugins
29
30 WP4 Modelling and Verification Plug-ins
31 D4.1 (M4.1) Definitions of plug-in tools
32 D4.2 (M4.2) Prototype plug-in tools
33 D4.3 Internal versions plug-in tools
34 D4.4 (M4.3) Public version of plug-in tools
35
36 WP5 Dissemination and Exploitation
37 D5.1 Initial dissemination report
38 D5.2 Proceedings of the first International workshop
39 D5.3 Proceedings of the second International workshop
40 D5.4 Dissemination and exploitation report
41 M5.1 Industry Interest Group (IIG) set up
42 M5.2 Active dissemination induces IIG extension
43 M5.3 RODIN platform supported by IIG
44
45 WP7 Project Review and Assessment
46 D7.1 Procedures for technical review and assessment
47 D7.2 Assessment report 1
48 D7.3 Assessment report 2
49 D7.4 Assessment report 3
50 M7.1 Setting up procedures for technical review and assessment
51 M7.2 First review: dissemination of material, assessment, analysis
52 M7.3 Second review: dissemination of material, assessment, analysis
53 M7.4 Third review: dissemination of material, assessment, analysis
54
D1.1 D1.2 D1.3 D1.4 D1.5 D1.6 D1.7
M1.1 M1.2 M1.3 M1.4
D2.1 (M2.1) D2.2 D2.3 (M2.2)
D3.1 D3.2 D3.3 D3.4 D3.5 D3.6
M3.1 M3.2 M3.3
D4.1 (M4.1) D4.2 (M4.2) D4.3 D4.4 (M4.3)
D5.1 D5.2 D5.3 D5.4A D5.4B D5.4C
M5.1 M5.2 M5.3
D7.1 D7.2 D7.3 D7.4
M7.1 M7.2 M7.3 M7.4
Project: RODIN
Date: Fri 16/04/04
Deliverable
Milestone
Workpackage
Result
Page 1

Furthermore, we demonstrate how the case studies will feed back the methodology tasks and development of plug-in tools in the following table:

	T2.1	T2.2	T2.3	T2.4	T2.5	T4.1	T4.2	T4.3	T4.4	T4.5
CS1	X		X	X		X	X		X	X
CS2	X	X	X			X		X	X	X
CS3	X	X	X	X	X	X	X	X	X	X
CS4	X	X	X		X	X	X	X		
CS5	X		X	X	X		X			X

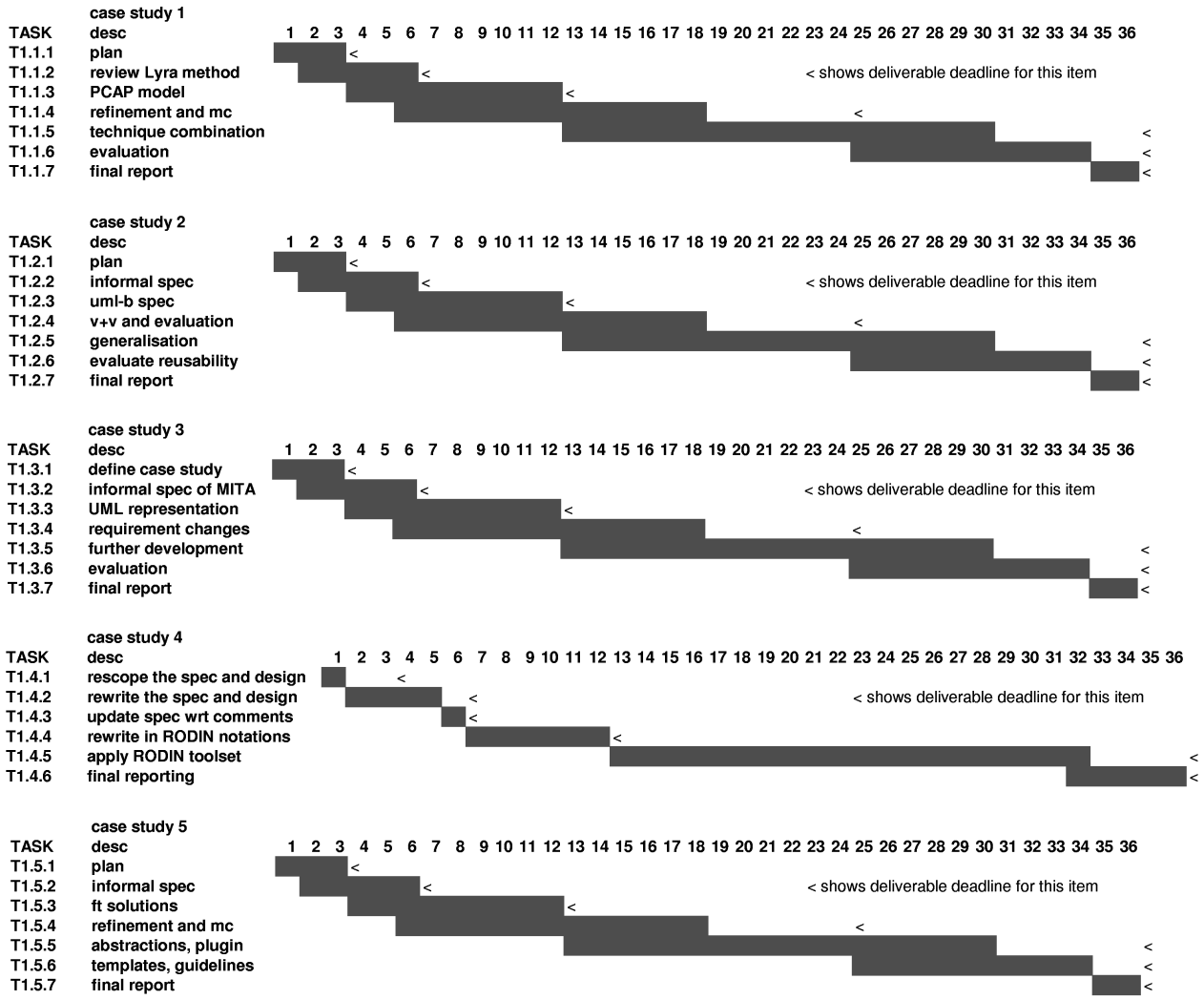
The table below shows how the tools and the open platform developed in WP3 and WP4 will contribute to the methodology tasks:

	T2.1	T2.2	T2.3	T2.4	T2.5
Open Tools Platform	X	X	X	X	X
UML – B Link	X	X	X	X	X
Model checking tools		X	X	X	
Model-based testing			X	X	X
Code Generation	X	X	X	X	X

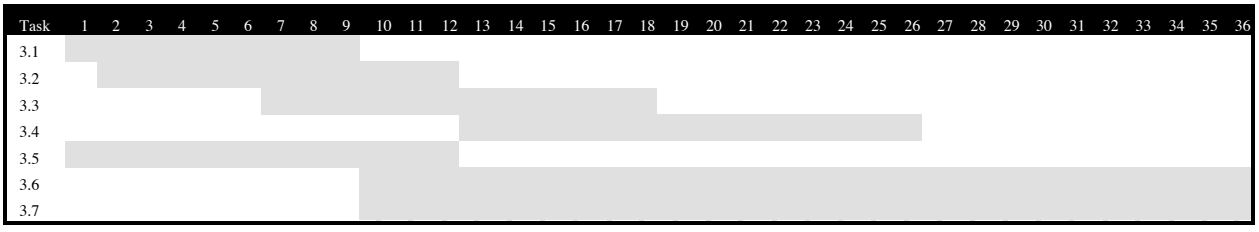
The table demonstrates that all of the results in the methodology tasks will be supported by and influenced by the tools kernel developed in WP3. The UML-B link will mean that the results on architectural design, generality and requirements traceability (T2.1, T2.2, T2.5) will be more accessible to design engineers and clearly these tasks will place requirements on features that should be supported by the UML to B translation tool developed in WP4. As well as being supported by the tools kernel, analysis of refinement, fault tolerant designs and adaptive architectures (T2.2, T2.3, T2.4) will be enhanced through the use of the model checking tools developed in WP4. The tools for model-based testing developed in WP4 will support the testing of systems developed using the architectural design approach (T2.1) and will need to take account of the results on generality (T2.2) and traceability (T2.5). Similarly the code generation tools will support code generation for systems developed using the architectural design approach (T2.1) and will need to take account of the results on generality (T2.2), mobility (T2.4) and traceability (T2.5).

Below we include the timetables that show the durations and the time periods of the tasks in WP1 and WP3 (in all other technical WPs all the tasks start at month 1 and finish at month 36).

WP1:



WP3:



Our position on RODIN work in WP3 and WP4 is as follows. The initial period will be devoted to thorough definition of the plug-ins and the kernel, writing their specifications which are open to the public, developing the general kernel architecture and plug-in interoperability standards. These results will be reported in the deliverables prepared within the first year (D4.1, D3.1, D3.2, D3.3). we are intentionally not planning to deliver any working prototypes by the end of year 1 because we believe it would undermine the

important phase of complex tool design and development. The first prototypes of the kernel (WP3) and plug-ins (WP4) will be developed by M18. Full working versions will be available by M30 with the final versions being ready by M36.

7.4. Workpackage list

Workpackage list (full duration of project)

Efforts shown are total of costed and uncosted. Uncosted (AC partners) effort figure shown ( )

Work-package No	Workpackage title	Lead contractor No	Person-months	Start month	End month	Deliv-erable No
WP1	Research drivers	2	221 (81)	0	36	7
WP2	Methodology	1	130 (8)	0	36	3
WP3	Open tool kernel	3	91 (4)	0	36	6
WP4	Modelling and verification plug-ins	8	95 (6)	0	36	4
WP5	Dissemination and exploitation	3	41 (6)	0	36	4
WP6	Project management	1	38 (12)	0	36	4
WP7	Project Review and Assessment	5	21 (4)	0	36	4
	TOTAL		639 (121)			

7.5. Deliverables list

Deliverables list (full duration of project)

Efforts shown are total of costed and uncosted. Uncosted (AC partners) effort figure shown ( )

Deliverable No	Deliverable name	WP No.	Lead Partic-ipant	Estimated person months	Nature	Dissemination level	Delivery date
D 1	D6.1 set up of project infrastructure including website	6	UNEW	5 (3)	O	PU	M1
D 2	D1.1 Definitions of case studies & evaluation criteria	1	AAU	9 (3)	R	PU	M3
D 3	D5.1 Initial dissemination report	5	ClearSy	7 (1)	R	PU	M3
D 4	D1.2 Traceable requirements documents for case studies	1	AAU	9 (3)	R	PU	M6

D 5	D3.1 Final decisions	3	ClearSy	11 (2)	R	PU	M6
D 6	D7.1 Procedures for Technical Review and Assessment	7	Praxis	6 (1)	R	RE	M6
D 7	D3.2 Event B language	3	ClearSy	11 (2)	R	PU	M9
D 8	D1.3 Initial report on case study developments	1	AAU	45 (15)	R	PU	M12
D 9	D2.1 Preliminary report on methodology	2	UNEW	45 (3)	R	PU	M12
D 10	D3.3 Specification of basic tools and platform.	3	ClearSy	15	R	PU	M12
D 11	D4.1 Definitions of plug-in tools	4	UoS	21 (3)	R	PU	M12
D 12	D5.4A Dissemination and exploitation report	5	ClearSy	6 (1)	R	PU	M12
D 13	D6.2A Project annual progress report	6	UNEW	11 (3)	R	CO	M12
D 14	D7.2 Assessment report 1	7	Praxis	5 (1)	R	RE	M12
D 15	D3.4 Prototypes of basic tools and platform	3	ClearSy	18	P	CO	M18
D 16	D4.2 Prototype plug-in tools	4	UoS	24 (3)	P	PU	M18
D 17	D5.2 Proceedings of first International workshop.	5	ClearSy	9 (1)	R	PU	M18
D 18	D1.4 Intermediate report on case study developments	1	AAU	45 (15)	R	PU	M24
D 19	D2.2 Intermediate report on methodology	2	UNEW	40 (2)	R	PU	M24
D 20	D5.4B Dissemination and exploitation report.	5	ClearSy	5 (1)	R	PU	M24
D 21	D6.2B Project annual progress report.	6	UNEW	11 (3)	R	CO	M24
D 22	D7.3 Assessment report 2	7	Praxis	5 (1)	R	RE	M24
D 23	D3.5 Internal versions of basic tools and platform	3	ClearSy	18	R	CO	M30
D 24	D4.3 Internal versions plug-in tools	4	UoS	25	P	PU	M30
D 25	D5.3 Proceedings of the second International; workshop	5	ClearSy	9 (1)	R	PU	M30

D 26	D1.5 Final report on case study developments	1	AAU	40 (20)	R	PU	M36
D 27	D1.6 Case study demonstrators	1	AAU	40 (15)	D	PU	M36
D 28	D1.7 Report on assessment of tools and methods	1	AAU	35 (10)	R	PU	M36
D 29	D2.3 Final report on methodology	2	UNEW	45 (3)	R	PU	M36
D 30	D3.6 Public versions basic tools and platform	3	ClearSy	18	D	PU	M36
D 31	D4.4 Public version of plug-in tools	4	UoS	25	D	PU	M36
D 32	D5.4C Dissemination and exploitation report	5	ClearSy	5 (1)	R	PU	M36
D 33	D6.2C Project annual progress report	6	UNEW	11 (3)	R	CO	M36
D 34	D7.4 Assessment report 3	7	Praxis	5 (1)	R	RE	M36

References

[Abrial et al 1980] J.-R. Abrial, S. A. Schuman, B. Meyer. Specification Language. In On the Construction of Programs, R.M. McKeag, A.M. Macnaghten (Eds.). Cambridge. pp. 343-410. 1980.

[Abrial 1996] J.-R. Abrial. The B-Book: Assigning programs to meanings. Cambridge University Press,1996.

[Back et al 1996] R.J.R. Back, A.J. Martin, and K. Sere, Specifying the Caltech asynchronous microprocessor. Science of Computer Programming 26, pp. 79-97, Elsevier. 1996.

[Back and Sere 1996] R.J.R. Back, K. Sere, From Action Systems to Modular Systems. Software - Concepts and Tools 17, pp. 26-39, Springer Verlag. 1996

[Butler 1996] M.J. Butler. Stepwise Refinement of Communicating Systems. Science of Computer Programming, Vol. 27, pp. 139-173, 1996

[Butler and Falampin 2002] M. Butler, J. Falampin. An Approach to Modelling and Refining Timing Properties in B. In Proceedings of RCS'02 - International workshop on Refinement of Critical Systems: Methods, Tools and Experience, France, January 2002.

[Butler et al 1996] M. Butler, E. Sekerinski, K. Sere. An Action System Approach to the Steam Boiler Problem. In J.-R. Abrial, E. Borger, H. Lang, (Eds), Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control, LNCS Vol. 1165. Springer-Verlag, 1996.

[Chessell et al 2002] M. Chessell, C. Griffin, D. Vines, M. Butler, C. Ferreira, P. Henderson. Extending the concept of transaction compensation. IBM Systems Journal, 41(4), p. 743-750. 2002.

[Cristian 1995] F. Cristian. Exception Handling and Tolerance of Software Faults. In M.R. Luy (Ed.) Software Fault Tolerance. 1995. pp. 81-108.

[Di Marzo and Romanovsky 2003] G. Di Marzo, A. Romanovsky. Designing Fault-Tolerant Mobile Systems. In Proceedings of the International Workshop on Scientific Engineering for Distributed Java Applications (FIDJI 2002), Luxembourg, 28-29 November 2002. N. Guelfi, E. Astesiano, G. Reggio. (Eds.). LNCS 2604, pp. 185-201. 2003

[Fuggetta et al 1998] A. Fuggetta, G.P. Picco, G. Vigna. Understanding code mobility. IEEE Trans. on Software Engineering, 24, 5, 242-361. 1998.

[Gaudel et al 2003] M.-C. Gaudel, V. Issarny, C. Jones, H. Kopetz, E. Marsden, N. Moffat, M. Paulitsch, D. Powell, B. Randell, A. Romanovsky, R. Stroud, F. Taiani. Final Version of DSoS Conceptual Model. DSoS deliverable CSDA1. (CS-TR: 782, School of Computing Science) University of Newcastle, July 2003.

[Hall 1996] A. Hall. Using Formal Methods to Develop an ATC Information System. IEEE Software, 13(2), pp. 66-76, March 1996.

[Hayes et al 2003] I. Hayes, M. Jackson, C. Jones. Determining the specification of a control system from that of its environment. In Formal Methods (Eds) K. Araki, S. Gnesi, D. Mandrioli. LNCS, 2805, pp.154-169. 2003

[Jones 1980] C.B. Jones. Software Development: A Rigorous Approach. Prentice Hall International. 1980.

[Jones and Shaw 1990] C.B. Jones, R.C.F. Shaw. Case Studies in Systematic Software Development. Prentice Hall International. 1990.

[Jones 1990] C.B. Jones. Systematic Software Development using VDM. Prentice Hall International. 1990.

[Jones et al 1991] C.B. Jones, K.D. Jones, P.A. Lindsay, R. Moore. A Formal Development Support System. Springer-Verlag. 1991.

[Jones et al 2002] C. Jones, A. Romanovsky, I. Welch. "A Structured Approach to Handling On-Line Interface Upgrades. In Proceedings of the 26th Annual International Computer Software and Applications Conference (COMPSAC 2002), Oxford, UK, 26-29 August 2002. pp. 1000-1005. 2002.

[Khomenko et al 2002] V. Khomenko, M. Koutny, W. Vogler. Canonical Prefixes of Petri Net Unfoldings. CAV 2002. 2000. (Full version: to appear in Acta Informatica.)

[Leuschel and Butler 2003] M. Leuschel, M. Butler. *ProB: A Model-Checker for B*. FM 2003: 12th Intl. FME Symposium. Pisa, September 8-14, Springer LNCS 2805, 2003.

[Meyer 1988-1997] B. Meyer. Object-Oriented Software Construction, 2nd edition, Prentice Hall, 1997 (first edition 1988).

[Meyer 2003] B. Meyer, *Towards Practical Proofs of Class Correctness*, in *ZB 2003: Formal Specification and Development in Z and B*, LNCS 2651, pp. 359-387, 2003.

[Milner et al 1992] R. Milner, J. Parrow, D. Walker. A Calculus of Mobile Processes (Parts I and II) Information and Computation 1992.

[Petre et al 2002] L. Petre, E. Troubitsyna, M. Walden. A Healthcare Case Study. In Proceedings of RCS'02 - International workshop on Refinement of Critical Systems: Methods, Tools and Experience, Grenoble, France, January 2002.

[PITAC 1999] PITAC - Report to the President. Information Technology Research: Investing in Our Future. President's Information Technology Advisory Committee. February 24, 1999. (www.hpcc.gov/pitac/report/)

[Randell 2000] B. Randell. Facing up to Faults (Turing Memorial Lecture). The Computer Journal, Volume 43, Issue 2, pp 95-106. 2000.

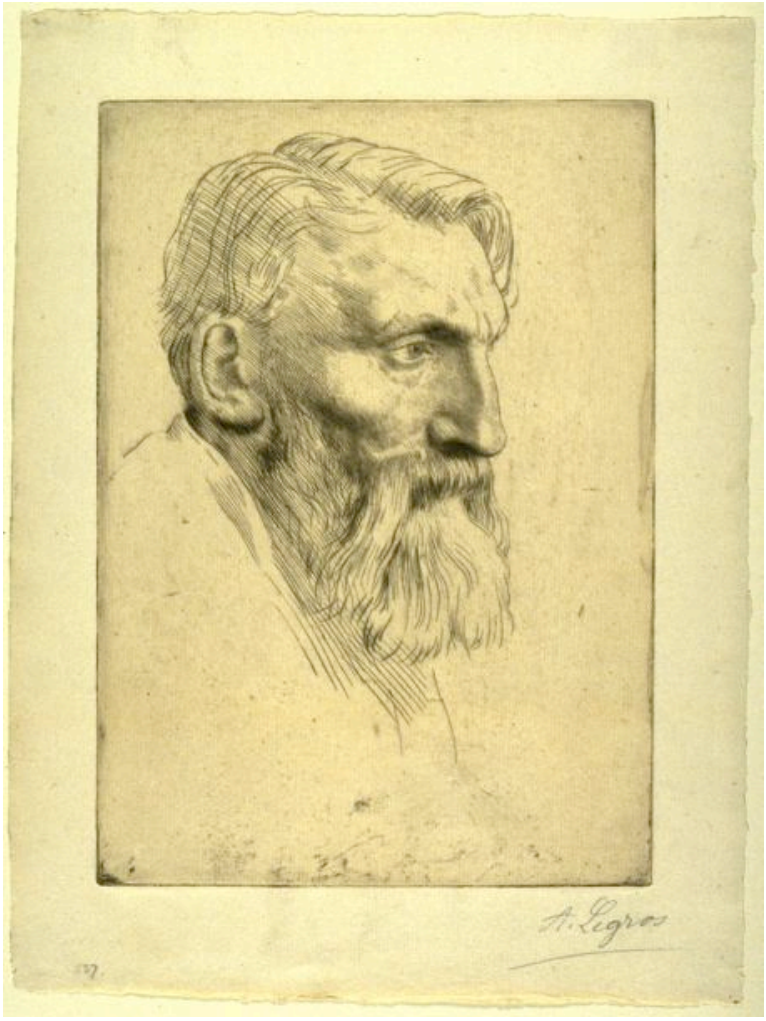
[Sere and Walden 2000] K. Sere, M. Walden. Data Refinement of Remote Procedures. Formal Aspects of Computing, Volume 12, No 4, December, pp. 278 - 297. 2000.

[Sere and Troubitsyna 1999] K. Sere, E. Troubitsyna. Safety Analysis in Formal Specification. In J. Wing, J. Woodcock, J. Davies (Eds.), FM'99 - Formal Methods. Proc. of World Congress on Formal Methods in the Development of Computing Systems, Toulouse, France, September, vol II, LNCS 1709, pp. 1564-1583, 1999

[Troubitsyna 2003] E. Troubitsyna. Developing Fault-Tolerant Control Systems Composed of Self-Checking Components in the Action Systems Formalism. In H.D. Van, Z. Liu (eds.). Proceedings of the Workshop on Formal Aspects of Component Software FACS'03, Pisa, Italy. TR 284, UNU/IIST, pp. 167-186. September, 2003

[Troubitsyna 2002] E. Troubitsyna. Integrating Safety Analysis into Formal Specification of Dependable Systems. In Proc. of Annual IEEE Workshop on Fault-Tolerant Parallel and Distributed Systems. Nice, France, April 2003.

[Xu et al 2002] J. Xu, B. Randell, A. Romanovsky, R.J. Stroud, A.F. Zorzo, E. Canver, F. von Henke. Rigorous development of an Embedded Fault-Tolerant System Based on Coordinated Atomic Actions. IEEE Trans. on Computers, 51, 2, pp. 164-179. 2002.



Alphonse Legros (French, 1837-1911), Portrait of Auguste Rodin
undated, Etching and drypoint on ivory laid paper.
University of Michigan Museum of Art, Museum Purchase 1976/1.227