

	Systèmes Embarqués et Grandes Infrastructures	Réservé à l'organisme gestionnaire du programme N° de dossier : ANR-08-XXXX-00 Date de révision :
	<b>Document de soumission B</b>	Edition 2008

Acronyme/ <i>Acronym</i>	U3CAT
Titre du projet/Proposal title (en français/ inFrench)	Unification des Techniques d'Analyse de Code C Critique
Titre du projet/Proposal title (en anglais/ in English)	<b>Unification of Critical C Code Analysis Techniques</b>

1. Programme scientifique et technique/Description du projet. *Technical and scientific description of the activities* .....5

1.1. Problème posé. Rationale. (1/2 page maximum).....5

1.2. Contexte et enjeux du projet. Background, state of the art, issues and hypothesis. (1 à 5 pages maximum).....5

1.3. Objectifs et caractère ambitieux/novateur du projet. Specific aims of the proposal, highlighting the originality and the novelty (1 à 2 pages maximum).....7

1.4. Positionnement du projet. Progress beyond the state of the art and relevance to the call for proposals (1 page maximum).....8

1.5. Description des travaux : programme scientifique et technique. Detailed description of the work. For each specific aim: a proposed workplan should be described (including preliminary data, work packages and deliverables). (10 pages maximum) .....10

1.5.1 Work Package 0: Management..... 10

1.5.2 Work Package 1: Floating Point Analysis..... 11

1.5.3 Work Package 2: Temporal Properties..... 13

1.5.4 Work Package 3: Combining Static Analysis Techniques..... 14

1.5.5 Work Package 4: Proof Management..... 16

1.5.6 Work Package 5: Trusting Formal Methods ..... 18

1.5.7 Work Package 6: Spreading Static Analysis ..... 20

1.6. Résultats escomptés et Retombées attendues. Expected results and potential impact. (1 à 2 pages maximum).....21

1.7. Organisation du projet. *Project management : structure and flow.* .....23

1.7.1 Relations between the Work Packages..... 23

TABLEAU des LIVRABLES et des JALONS (le cas échéant)/ *Deliverables and milestones* ..... 25

1.8.	Organisation du partenariat. Description of the Consortium.....	27
1.8.1	Pertinence des partenaires. Presentation of the relevance of each partner to the proposal. ....	27
	Airbus France [AIF] .....	27
	ATOS Origin [ATOS] .....	27
	CEA List – Software Reliability Laboratory [CEA] .....	28
	CNAM Cedric laboratory [CNAM].....	28
	CS Communication & Systèmes [CS].....	28
	Dassault Aviation [DA] .....	29
	Hispano-Suiza [HISP] .....	29
	INRIA project-team Gallium [GALLIUM].....	30
	INRIA Project-team LANDE [LANDE].....	30
	INRIA Project-team PROVAL [PROVAL] .....	31
1.8.2	Complémentarité des partenaires. Description of complementarity within the consortium. ....	31
1.8.3	Qualification du coordinateur du projet. <i>Principal investigator: skills and CV.</i> .....	32
1.9.	Stratégie de valorisation et de protection des résultats. Data management, data sharing, intellectual property strategy, and exploitation of project results. (1 page maximum) .....	32
2.	Justification scientifique des moyens demandés. <i>Requested budget : detailed financial plan.</i> ....	33
2.1.	AIRBUS France.....	33
2.1.1	Large equipment.....	33
2.1.2	Personnel. ....	33
2.1.3	Services, outward facilities. ....	33
2.1.4	Travels.....	33
2.1.5	Expenses for inward billing.....	33
2.1.6	Other expenses. ....	33
2.2.	ATOS Origin .....	33
2.2.1	Large equipment.....	33
2.2.2	Personnel. ....	33
2.2.3	Services, outward facilities. ....	33
2.2.4	Travels.....	33
2.2.5	Expenses for inward billing.....	33
2.2.6	Other expenses. ....	34

2.3.	CEA List .....	34
2.3.1	Large equipment.....	34
2.3.2	Personnel.....	34
2.3.3	Services, outward facilities.....	34
2.3.4	Travels.....	34
2.3.5	Expenses for inward billing.....	34
2.3.6	Other expenses.....	34
2.4.	CNAM-CEDRIC laboratory.....	34
2.4.1	Large equipment.....	34
2.4.2	Personnel.....	34
2.4.3	Services, outward facilities.....	34
2.4.4	Travels.....	34
2.4.5	Expenses for inward billing.....	35
2.4.6	Other expenses.....	35
2.5.	CS .....	35
2.5.1	Large equipment.....	35
2.5.2	Personnel.....	35
2.5.3	Services, outward facilities.....	35
2.5.4	Travels.....	35
2.5.5	Expenses for inward billing.....	35
2.5.6	Other expenses.....	35
2.6.	Dassault Aviation.....	35
2.6.1	Large equipment.....	35
2.6.2	Personnel.....	35
2.6.3	Services, outward facilities.....	35
2.6.4	Travels.....	36
2.6.5	Expenses for inward billing.....	36
2.6.6	Other expenses.....	36
2.7.	Hispano-Suiza.....	36
2.7.1	Large equipment.....	36
2.7.2	Personnel.....	36
2.7.3	Services, outward facilities.....	36
2.7.4	Travels.....	36

2.7.5	Expenses for inward billing.....	36
2.7.6	Other expenses. ....	36
2.8.	INRIA-Gallium project.....	36
2.8.1	Large equipment.....	36
2.8.2	Personnel. ....	36
2.8.3	Services, outward facilities. ....	36
2.8.4	Travels.....	37
2.8.5	Expenses for inward billing.....	37
2.8.6	Other expenses. ....	37
2.9.	INRIA-LANDE project. ....	37
2.9.1	Large equipment.....	37
2.9.2	Personnel. ....	37
2.9.3	Services, outward facilities. ....	37
2.9.4	Travels.....	37
2.9.5	Expenses for inward billing.....	37
2.9.6	Other expenses. ....	37
2.10.	INRIA-Proval project.....	38
2.10.1	Large equipment.....	38
2.10.2	Personnel. ....	38
2.10.3	Services, outward facilities. ....	38
2.10.4	Travels.....	38
2.10.5	Expenses for inward billing.....	38
2.10.6	Other expenses. ....	38
Annexes.....		38

# **1. Programme scientifique et technique/Description du projet.** ***Technical and scientific description of the activities***

## **1.1. Problème posé. Rationale. (1/2 page maximum)**

Embedded software is increasingly prevalent in everyday life. Moreover, very large applications (with hundreds of thousand lines of code when not millions) are now commonly embedded in various objects and are often used to perform critical tasks, for which a failure would result in serious economic loss or even casualties. This is in particular the case in aeronautics and in the automotive industry, where safety concerns are very high. In addition, as software grows in size, it becomes difficult to ensure that testing techniques provide an adequate coverage of all situations that may occur during the life of the system. There is thus a strong need to provide the architects of such systems with adequate tools that allow them to state precisely the properties that the application must meet and to verify statically that the implementation really respects them. In this context, targeting the analysis of C code is very relevant, as a lot of programs in the embedded world are either written directly in C, or written in an higher-level language which uses C as an intermediate language during the compilation process (*e.g.* Scade).

There is no “silver bullet” analysis method which would be able to tackle every programming and verification issue. Various analysis techniques, such as model-checking, abstract interpretation and weakest preconditions calculus all have their pros and cons. It is thus desirable to provide a unified interface that would allow to perform several analyses on the same code and to let these analyses cooperate in order to prove properties that could not have been established by one single technique. This is the main scientific and technical challenge that will be addressed by the U3CAT project. It will be built upon the results of the RNTL 2005 CAT project, which has been very successful in delivering the Frama-C (<http://www.frama-c.cea.fr>) platform for the analysis of C programs and the ANSI/ISO C Specification Language (ACSL), both of which form the grounds upon which U3CAT components will be built. Additionally, the openness of the Frama-C platform makes it possible to integrate new static analysis techniques as they reach maturity.

Many subjects investigated in U3CAT are just reaching the point of industrial applicability, but the developments will be led by actual issues raised by the industrial partners of the project, so that U3CAT qualifies as an **industrial research platform project**. Such a partnership has proved to be very stimulating during the CAT project. Another challenge of U3CAT is to foster the use of static analysis in industrial software development cycles. This will be addressed with a close collaboration between tool providers, industrial final users and IT service companies of the critical software world.

## **1.2. Contexte et enjeux du projet. Background, state of the art, issues and hypothesis. (1 à 5 pages maximum)**

Together with the growing importance<sup>1</sup> of embedded systems for society and economy, requirements for their correctness are expressed more and more formally through various normative documents such as the DO-178B and the forthcoming DO-178C in avionics, IEC 61508 in automotive, Cenelec EN50128 for railways and ESA PSS-05 for space. Traditional techniques and in particular manual code review will not scale up and will not be able to provide the needed guarantees on the vast amount

---

<sup>1</sup>Information Society Technologies. Embedded Systems – Facts and Figures, [http://cordis.europa.eu/ist/embedded/facts\\_figures.htm](http://cordis.europa.eu/ist/embedded/facts_figures.htm)

of code that is expected to be produced by the embedded software industry in the near future. Even current testing-based methods might prove to be too costly for the complex software that will need to be checked. There is thus a strong need to propose innovative and more cost-efficient analysis techniques that can provide the desired safety, security and functional properties with an appropriate confidence level. More precisely, the formal verification market is expected to reach 110 millions € in 2008, with an annual growth rate around 20%<sup>2</sup>.

In this context, we can classify the available tools into the following main categories:

- 1) Tools based on abstract interpretation. Abstract interpretation has been introduced in the late 70s. Its main idea is to compute results over an abstract domain which reflects the concrete values manipulated by the analyzed program. Depending on the degree of abstraction, it is possible to assess more or less precise properties, at the expense of huge computation time and memory costs for the more detailed domains. In particular, Polyspace Verifier (<http://www.mathworks.com/products/polyspace>) is a widely deployed tool which uses abstract interpretation to verify the absence of run-time errors (such as an invalid pointer access or a division by 0). It aims at being fully automated, which is crucial for its use in production, but automation comes at a cost. Indeed, it tends to use quite generic abstract domains, which leads sometimes to too coarse approximations and thus to false alarms (*i.e.* the tool cannot guarantee the absence of error at run-time). On the other hand, the Astrée tool (<http://www.astree.ens.fr>) focuses on a particular kind of C code, namely Scade-generated C code. This specialization allows Astrée to obtain remarkably precise results on such codes, with nearly no false alarm. However, like Polyspace, Astrée concentrates on the detection of run-time errors, and offers very limited support to verify behavioral properties.
- 2) Weakest preconditions calculus (WP). Caduceus (<http://caduceus.lri.fr>) proposes an expressive specification language to state properties that need to be formally verified. Based on Hoare logic, it transforms these specifications into first-order formulas that can then be discharged by various theorem provers (automated like Alt-Ergo or interactive like Coq). However, it uses a quite abstract memory model which imposes some limits on the kind of programs that can be analyzed by Caduceus (in particular, casts are not allowed). Moreover, it demands an important specification effort to complete a proof, since Hoare logic imposes to write invariants for each loop in the analyzed code, and to specify the pre- and post- conditions of each function called. Parts of this information could be retrieved through abstract interpretation. This is why the Caveat tool (<http://www-list.cea.fr/labos/gb/LSL/caveat/index.html>) attempts to combine some very basic abstract interpretation techniques with a Hoare logic. However, Caveat targets only a fragment of the C language. In particular, pointer aliases are not supported. This makes the tool unsuitable for complex software.
- 3) Model-checking. A model checker attempts to verify a property, expressed as a logical formula by extracting a model of the analyzed system and proving that this model satisfies the formula. While these techniques have been extensively used for the formal verification of hardware and concurrent systems, few tools target C programs. Indeed, formulas resulting from the model of a program are often very difficult to handle automatically. Similarly,

---

<sup>2</sup>Fast Gmbh and Technical University of Munich. Study of Worldwide Trends and R&D Programs in Embedded Systems in View of Maximizing the Impact of a Platform in the Area. 2005

resulting models are in general very large and intractable in practice without using specific means to avoid “state explosion” issues. A few tools attempt to deal with these problems. This is in particular the case of BLAST (<http://mtc.epfl.ch/software-tools/blast>), which uses model checking to verify that software respects the behavioral properties of the libraries it depends upon. Similarly, the SLAM project (<http://www.research.microsoft.com/slam>) shows how these techniques can be applied to hardware drivers.

While these three main categories of tools have historically evolved quite separately, recent work tends to incorporate techniques from various origins. We have already mentioned that Caveat used some primitive abstract interpretation to relieve the user of writing some annotations that would be needed in a pure Hoare logic-based setting. More recently, an extension to BLAST has been devised, which allows to incorporate a more or less important level of abstract interpretation into the model checker. Last, the Frama-C framework, developed jointly by CEA-LIST, INRIA, Airbus France, Dassault Aviation, Siemens VDO (now Continental), and France Télécom, is dedicated to the collaboration of various analysis techniques and tools but does not cope yet with all kinds of questions that arise in embedded software verification.

### **1.3. Objectifs et caractère ambitieux/novateur du projet. Specific aims of the proposal, highlighting the originality and the novelty (1 à 2 pages maximum)**

As it has been said above, proving safety and security properties as well as functional properties on a large software development can only be attained by the use of various techniques, and furthermore, by the collaboration of these techniques. There is thus a need for a unified platform that would offer a single entry point for the various tools involved in the validation of a given software project. The Frama-C framework is such a platform. U3CAT aims at discovering and implementing extensions of this framework in order to broaden the use of static analyzers in the embedded software industry. In order to achieve this goal, several important issues must be addressed. These issues can be summarized in three main categories. A first point consists in enhancing the Frama-C framework, so that it can handle efficiently new kinds of code and/or properties. Second, existing analyses within the framework must be hardened, allowing them to analyze more code faster, and to provide a better interaction with the user and with other analysis. Last, during the validation of a critical embedded system, one has to convince a certification authority of the quality of the tools used all along the development (and verification) chain. The Frama-C framework must thus provide relevant elements to prove that its results can be trusted. In the remaining of this section, we detail how these three points will be reflected in U3CAT.

First, the framework must be enhanced to deal with specific domains and properties. This includes in particular on the one hand the ability to analyze precisely floating point computations, and on the other hand the possibility to express some relatively simple temporal properties and automata-like properties. These domains were beyond the scope of the CAT project, whose main focus was the set-up of the Frama-C framework, and as such dealt primarily with the most fundamental C constructs and properties. Nevertheless, it is crucial to properly handle both of them, as most critical embedded systems incorporate these notions in some way. Some tools already exist for both domains, such as Gappa (<http://lipforge.ens-lyon.fr/www/gappa/>) for floating point analyses and Spin (<http://www.spinroot.com/>) or CADP (<http://www.inrialpes.fr/vasy/pub/cadp.html>) for temporal properties. However, these tools are usually dedicated to a particular task. The integration of dedicated analyses in a larger verification environment remains largely to be done.

Concerning the hardening of existing Frama-C components, a first issue is to augment the collaboration level between the various analyses offered by the Frama-C framework, and to facilitate the communication between the Frama-C analyses and external, more specialized analyzers. Existing Frama-C tools collaborate mostly *via* the exchange of generated ACSL formulas. In other words, this is a quite shallow binding. Deeper bindings, in particular for the collaboration between abstract interpretation and deductive methods, though scientifically challenging, are very promising to augment the verification power of the tool. Another issue in this area concerns the feedback that the system can give to the user, and the possibilities offered to overcome the limitations of automated approaches. Indeed, even if a very good level of automation can be attained, the small percentage of unresolved proof obligations can result in hundreds or thousands of remaining goals for large systems, where the initial number of proof obligations can be counted in hundreds of thousands. Thus, an efficient proof management system must be devised in order to cope with this huge amount of proof obligations.

Last, development and validation of critical embedded software is often constrained by normative documents. In this context, it is important to investigate what is needed in the development of the Frama-C framework to favor the qualification of the framework as a verification tool for a given software. This includes in particular establishing formally the properties of some of the most fundamental parts of the framework. This approach follows an increasing trend in the programming language and static analysis communities, as proof assistants become more mature. As exemplified by the development of the CompCert C compiler, formally proving the properties of a real-sized tool in this area remains a very difficult task, but is at least doable. Moreover, such an effort could be used to gain confidence over the whole tool chain going either from C to object code or from higher-level languages to C. Two approaches are possible in this area. On the one hand, we will investigate the link between the semantics of static analyzers and of certified compilers. On the other hand, we will investigate how to extract ACSL specifications from higher-level models, such as Scade specifications. These ACSL formulas will in particular allow to prove the correctness of the bindings between generated code and external libraries.

Moreover, U3CAT aims at providing tools that can be integrated in industrial software development cycles. This is a very ambitious objective that cannot be reached only through purely scientific and technical breakthrough. Industrial methodologies will be studied in order to understand how and where static analysis methods can be deployed so as to maximise the ratio between safety and development costs.

#### **1.4. Positionnement du projet. Progress beyond the state of the art and relevance to the call for proposals (1 page maximum)**

The main research area of U3CAT inside the ARPEGE call for projects is clearly the topic 5 concerning safety, security and related tools (Sûreté, sécurité, et outils associés). Topic 4 (Méthodes et outils logiciels de spécification, modélisation, validation et optimisation/Specification, validation, modelisation and optimization methods and software tools) is a secondary target for U3CAT, at least for specification and validation tools.

U3CAT relies heavily on the results of the 2005 RNTL project CAT. CAT provided the basic infrastructure and the fundamental analyses that are needed to perform validation and verification tasks on C code. U3CAT proposes to follow this direction by greatly extending the kind and amount of code and properties the Frama-C framework can deal with. Besides that, a certain number of research projects, both at European and French level, propose interesting approaches for static

U3CAT

analysis techniques and aim at spreading their usage in the software industry. The most closely related to U3CAT objectives are the following.

- The 2007 ITEA2 **ES\_PASS** project aims at promoting the use of static analysis techniques for software verification. With respect to U3CAT, it is more orientated toward methodological aspects, such as the insertion of a given tool into an existing software development process and the place of static analysis tools in normative documents and in front of certification authorities.
- The German project **AVACS**, whose second phase started in 2008, investigates the mathematical analysis of complex software systems. Its primary domains of interest are quite different than the one of U3CAT, as it is more orientated toward temporized systems and hybrid automata. Moreover, it is more interested in the modeling of complex systems and the verification of high-level models than in specification and validation of C code. In other words, its primary target is a bit higher than the one of U3CAT in the traditional software development cycle.
- The ANR **CERPAN** project deals with formal verification of properties on floating-point computation. On the contrary to U3CAT, it does not target specifically safety critical C programs but rather numerical analysis programs. Moreover it does not aim at integrating its techniques into a larger framework such as Frama-C. It is however expected that the outcome of this project will form the basis upon which Frama-C floating-point tools will be built.
- The 2004 ANR ACI Project **APRON** (<http://apron.cri.enscm.fr/library>) aimed at defining a common interface for numerical abstraction domains, and to provide implementations relative to this interface for many existing domains. APRON's objective is not to provide a complete source code static analyzer, but an essential building block of such a tool. The U3CAT proposal is to use (as a black box) the library produced in the APRON project in its analysis framework. In this way, the project U3CAT will benefit all the future improvements to the APRON library.
- **ASTREE** is a static analyzer that was supported in part by the 2002 exploratory ANR RNTL project of the same name. The ASTREE analyzer has been very successfully used for the verification of critical software in the avionics field. Compared to the Frama-C analysis framework produced in the CAT project and whose extension is proposed in U3CAT, ASTREE relies solely on abstract interpretation techniques, whereas Frama-C aims at allowing the collaboration of various analysis techniques. Additionally, ASTREE is not publicly available as of March 2008, whereas Frama-C is an Open Source project. The intention with the Open Sourcing of Frama-C is to lower the barrier to entry for the integration of static analysis techniques into industrial production as much as possible. A lower barrier to entry makes it possible to explore the use of static analysis tools for industrial actors in fields where reliability is not yet mandated by strict certification norms, such as the automotive industry.
- The proposed ANR project **DECERT** is dedicated to the enhancement of automated decision procedures in three areas that are extremely relevant for C code verification, namely the expressivity of the decision procedures (*i.e.* the kind of formulas they can handle), their efficiency, and their ability to produce certificates when they find a proof. The Frama-C framework is a natural client for these decision procedures, and DECERT plans to integrate

them in Frama-C, but this work is orthogonal to the proof management system that is envisaged in U3CAT.

**1.5. Description des travaux : programme scientifique et technique. Detailed description of the work. For each specific aim: a proposed workplan should be described (including preliminary data, work packages and deliverables). (10 pages maximum)**

The main objective of U3CAT is to support wide adoption of static analysis techniques in the embedded software world. In order to reach this objective one has to drastically improve the existing tools and to ease their usage. Concretely, U3CAT will enhance, harden and promote the Frama-C framework that has been developed during the 2005 RNTL Project CAT. The project will be divided in five technical work packages (WP), one management work package and one dissemination work package. The work packages fall either

- in the enhancement category (WP 1 Floating Point Analysis and WP 2 Temporal Properties). These work packages address the problem of the verification of properties in two specific domains that were not taken into account in the Frama-C framework but are fundamental for embedded software;
- or in the hardening category (WP 3 Combining Static Analysis Techniques, WP 4 Proof management, WP 5 Trusting Static Analysers). The methodological framework has to be hardened in the sense that whenever static analysis becomes too costly or even impossible to perform an alternative has to be available. The tools have to be hardened in the sense that the users must have evidences of their correctness;
- or in the promotion category (WP 6 Spreading Static Analysis). The Frama-C framework is Open Source software. As such, creating a community of users and a community of developers is critical for its future. Both of these communities must be actively supported to spread static analysis tools.

For each work package, the list of participants is given (the work package leader being in bold case) together with the number of person.month per partner.

**1.5.1 Work Package 0: Management**

- Participants: **CEA(6)**
- Goal

WP 0 goals are

1. To ensure that all actions are performed correctly and within the rules and regulations established by the ANR and in the consortium agreement including financial and legal management;
2. To ensure that the work and tasks are performed on time, within budget and to the highest quality;
3. To ensure that periodic reporting is performed in the most efficient and pragmatic way according to ANR guidelines, and to inform all members of the consortium of all important and impacting information that can influence the outcome of the project;

4. To permit knowledge management through internal web sites and mailing lists
- Detailed Work Plan

The U3CAT collaborative process set-up shall include the following activities:

1. To organise an initial seminar for compiling, discussing, refining and sharing relevant information and experience (engineering, research and technology, methodology).
  2. To plan and organise regularly U3CAT plenary meetings (at least twice a year)
  3. To generate the U3CAT evaluation protocol
  4. To prepare and organise the project official reviews
- Methods and Technical Solutions

The project manager generates the template of the final report and requests final contributions from the partners (achieved results). The partners perform a synthesis of their results and generate the relevant sections of the report. They support the final report integration. All participants prepare their contribution to the final presentation.

- Risk Analysis and Mitigation Plan

The project manager has already successfully managed this kind of project. Therefore risk of failure of this work package is negligible and does not need any mitigation plan.

#### **1.5.2 Work Package 1: Floating Point Analysis**

- Participants: AIF(6), CEA(14), DA (4), LANDE(27), HISP (3.6), PROVAL(4)
- Goal: Adapt static analysis and verification tools to deal accurately with floating-point computations. Using floating-point numbers to represent real numbers often yields to introduce subtle flaws in embedded C programs. In fact, many mathematical properties holding over the real numbers are compromised on floating-point numbers (for example, associativity of addition is lost). Consequently, the conformity of a computation with respect to the expected mathematical result is no longer guaranteed, which is unacceptable for C programs embedded in critical systems. In addition, although the IEEE 754 standard was introduced to specify how to implement basic floating-point operations, the result of a floating-point expression still depends on the features of the hardware computation unit. For example, the result of ternary expression  $a + b + c$  depends on the processor architecture in use, as IEEE 754 does not specify the format of intermediate computations (Intel performs intermediate computations on 80 bits while Sparc performs on 64 bits). The primary goal of this work package is to implement an IEEE-754 conformant abstraction for floating point numbers that copes with all the basic operations (add, subs, mult, ...), all the formats (single, double and both extended), all the four rounding modes (to-the-nearest, to-zero, up, down) and to adapt our static analysis and verification tools to deal accurately with basic floating-point computations. We do not forecast to address in this project all the problems related to floating-point computations, but rather to focus on designing suitable abstractions for the IEEE standard and adapting the abstract domains of Frama-C on floating-point numbers. Our secondary goal in this work package is to understand how floating-point computations are performed in industrial C code and to put at work our static analysis and verification tools on

these programs. In the context of the whole project, this work package has a strong connection with WP3 (Combining Static Analysis Techniques) in the sense that the floating-point specific analyses devised in WP1 will have to be combined with the other ones. In addition, as for all the other technical work packages, the availability of new analyzers will greatly help the dissemination of the Frama-C platform as envisaged in WP6 (Spreading Static Analysis).

- Detailed Work Plan

1. Industrial specification: industrial partners provide documents and source code to explain precisely how they perform floating-point computations and what kind of properties and code they wish to verify. In particular the status of programs manipulating formats, rounding modes, infinities and NaN symbols, floating-point exceptions shall be decided. The problem of unspecified use of transcendental functions shall also be discussed (libraries, correct rounding...) although this is clearly outside the scope of the work package.
2. Abstract Domains adaptation: modify existing floating-point abstract domains to deal accurately with IEEE-754 compliant floating-point computations in order to improve the value analysis of Frama-C and a feasible path analysis implementation. Put at work our improved static analysis tools on C programs provided by our industrial partners.
3. Deductive verification: propose memory models dealing with floating point numbers following the industrial specification. Starting from the memory model designed in Caduceus, we aim at enhancing automation of these methods and to integrate them in Frama-C.
4. Assessment: some experiments will be made, based on characteristic C functions, and previous experimentations realized by industrials on other floating point analysis tools (for complementary aspects).

- Methods and Technical Solutions

The WP leader will provide a template to industrial partners to collect all the necessary information. Item 2 will provide an extension of the value analysis of the Frama-C framework developed during the CAT project and an implementation of feasible path analysis for C programs that exploit abstract lattices operations over floating-point numbers. Feasible path analysis is a precise static analysis aiming at detecting dead code. It is particularly useful in the context of DO-178B certification as it helps removing dead code but also finding input data that activates a given path. Implementing such an analysis in the presence of floating-point computations is particularly challenging as it requires developing specific operators over abstract lattices on floating-point numbers. Item 3 will first define a first order memory model for floating point arithmetic, replicating the work initiated in Caduceus, so that deductive verification can be performed on programs involving floating point numbers in Frama-C. The resulting proof obligations will then be handled using interactive or automated theorem provers, or a combination of both. Interactive proof assistants such as Coq, PVS or HOL-light already have formalizations of IEEE-754 floating point numbers, which will be used to realize the model used to build the proof obligations, allowing interactive verification. Automated theorem provers supporting real numbers (such as Z3 or CVC3) will also be used to discharge some of the proof obligations, using a suitable axiomatisation of floating point operations with real numbers. Finally, the tool Gappa (<http://lipforge.ens-lyon.fr/www/gappa/>) will be used to

discharge the proof obligations related to constant intervals, and will be integrated to the Coq proof assistant to combine interactive and automated verification.

All items shall be validated by the industrial partners according to their initial specification.

- Risk Analysis and Mitigation Plan

Item 1 relies on the capacity of industrial partners to synthesise information: this has already been successfully done in former projects with these partners and they have already expressed strong needs in this area. Items 2 and 3 are very prospective. Each of them may fail depending on the level of the industrial requirements. In any case, academic literature addresses some of these problems, even if it is not for the C language, and convincing prototypes will be implemented addressing the most significant industrial needs. The rest of the project would not be significantly impacted by the failure of any item of this work package.

### 1.5.3 Work Package 2: Temporal Properties

- Participants: AIF(6), ATOS(5), CEA(24), DA (7), HISP(3.6), PROVAL(15)

- Goal

Many properties that one wishes to verify in embedded software include some kind of temporal aspect. This stems from the fact that embedded software are often structured as reactive programs, i.e. as one global infinite loop defining a cyclic behaviour and reacting to sensors inputs. For example control/command programs have such a structure. Such code may be partially generated from high level specifications. To tackle cyclic behaviours of programs we propose to extend the ANSI/ISO C Specification Language (ACSL) with temporal operators like the “pre” of Lustre. Then we will propose techniques to perform static analysis geared toward verifying this new kind of properties. For the same reasons than in the preceding work package, this work package is connected with WP3 (Combining Static Analysis Techniques) and WP6 (Spreading Static Analysis). In addition, temporal properties are very often present in Scade specifications that are to be investigated formally in WP5 (Trusting Formal Methods), so that this item of WP5 strongly depends on WP2.

- Detailed Work Plan

1. Industrial specification: Industrial partners provide documents and source code to explain precisely what kind of properties and code they wish to verify. In particular properties referring to multiple cycles in the past and properties of basic temporal operators will be detailed.
2. Extension of ACSL: The basic blocks to define properties with temporal operators shall be added to ACSL and defined formally. First the notion of “cycle” will be defined. Second the specification of properties depending on multiple cycles in the past must be eased. Automata-like specification will be studied. These blocks must fulfil the industrial requirements.
3. Adaptation of Abstract Interpretation and Deductive Methods: Logical constructs introduced in item 2 will be taken into account in the existing Frama-C value analysis and weakest precondition plug-ins.
4. Integrating Model Checking Ideas within Abstract Interpretation: Whereas abstract interpretation merges the states of a program to limit state explosion, model checking

typically keeps the states separated. To deal with temporal properties and automata-like properties abstract interpretation should refrain from merging some states of the program. To manage large amounts of separate states model checking techniques have been proved to be very efficient<sup>3</sup>. This task will implement the model checking algorithms used by modern model checkers in the abstract interpretation plug-in of Frama-C.

5. Predicate abstraction: Predicate abstraction is a technique which combines theorem proving and model checking to prove properties of infinite state systems. In some sense, it is another technique for discovering useful inductive invariants on programs, in order to perform static verification. It has been successfully applied in tools like SLAM/SDV (Microsoft, driver verification) and BLAST (Berkeley U., which is based on CIL like Frama-C). We will study how to integrate the ideas of predicate abstraction into the Frama-C framework.
  6. Assessment: To illustrate the capability of the solutions implemented in this WP, some industrial application extracts will be used. For instance, these extracts will be parts of control-command synchronous automata, generated mostly automatically (from formal specifications) but containing also some functions developed manually in ANSI C.
- Methods and Technical Solutions

The WP leader will provide a template to industrial partners to collect all the necessary information. Item 2 will lead to a new version of the ACSL official design document. Item 3 will be implemented either as extensions of the value analysis and weakest pre-condition analysis of the Frama-C framework developed during the CAT project or as a conservative specification translation to a non-temporal ACSL specification. Item 4 will be an extension of the value analysis plug-in of Frama-C. Item 5 will be a new plugin.

All items shall be validated by the industrial partners according to their initial specification.

- Risk Analysis and Mitigation Plan

Item 1 has no specific risk according to the industrial partners' previous experience in temporal specification and programming. Item 2 shall not present any risks as many other languages like Lustre successfully defined such operators. Item 3, 4 and 5 are research activities. Their failure is not critical to the other parts of the project.

#### 1.5.4 Work Package 3: Combining Static Analysis Techniques

- Participants: AIF(4), CEA(24), CNAM(2), CS(6), DA (1), HISP(3.2), **PROVAL(7)**
- Goal

Frama-C is a collaborative framework for static analysis techniques that has been implemented during the CAT project. This work package is dedicated to the study of some possible promising combinations between such techniques or variation of such techniques. In particular, the collaboration between abstract interpretation and weakest-precondition computation will be

---

<sup>3</sup> Clarke, Grumberg and Peled. Model Checking. *MIT Press Jan. 1999.*

extensively studied and implemented. Slicing and impact analysis techniques will also be combined with weakest-precondition computation in order to specialise the source code according to specific properties. This will eventually ease or even make possible a deductive proof by removing spurious C constructs that could impair the precondition computation in a correct automatic manner. Different kinds of source code require heterogeneous levels of abstractions for weakest-precondition computations. Therefore proper memory models must be defined and be combined in order to cope with large source code. To improve abstract interpretation in itself, the collaboration between non relational and numerical relational lattices will be implemented in a generic way. As said above, this work package will in particular take into account the results of WP1 (Floating Point Analysis) and WP2 (Temporal Properties). In addition, like these two work packages, an efficient collaboration between the analyzers will lower the cost of static analysis and thus help its dissemination, as planned in WP6 (Spreading Static Analysis). Last, the formalisation effort of WP5 (Trusting Formal Methods) will be primarily based on the models and abstractions devised during WP3.

- Detailed Work Plan

1. Combining Abstract Interpretation and Weakest-Precondition: The main challenge is to use abstract interpretation as a path sensitive and context sensitive analysis to generate properties influencing the weakest-precondition assignment and call deduction rules. We seek in particular properties of memory separation. The combination between path and context *insensitive* analysis (typing rules) and weakest precondition computations has been successfully implemented<sup>4</sup>.
2. Combining Slicing and Impact Analysis with Weakest Precondition: The objective of this task is to specialise source code with respect to its logic properties and its abstraction level before performing weakest precondition computations. Impact analysis will be used to find which proof obligations should be regenerated whenever the set of code annotations is modified.
3. Combining Memory Model for Weakest Precondition: Low-level models will be implemented to allow deductive verification on low-level source code *i.e.* on source code with casts and unions. Such a low-level memory model will be based on the memory model used by the certified compiler CompCert.
4. Combining Abstract Interpretation Lattices: Interfacing the Apron library with the Frama-C abstract interpretation plug-in.

- Methods and Technical Solutions

All items shall be specified by source code, properties and verification goals examples. All items shall be validated by the very same industrial partners according to these initial verification goals. The first item will be studied by designing a weakest-preconditions assignment rule depending on the program point. This has been already partly implemented in the Caveat tool developed by the CEA. The second item will propose a methodology together

---

<sup>4</sup> [Thierry Hubert](#) and [Claude Marché](#). Separation analysis for deductive verification. In *Heap Analysis and Verification (HAV'07)*, Braga, Portugal, March 2007.

with a user interface implementing this methodology. The existing slicing and impact analysis plug-ins of Frama-C will be extended with new criteria geared toward preservation of the semantics of annotations. Item 4 deals with interfacing the APRON library and Frama-C. APRON is a library offering various numerical (relational) abstract domains with a single unified interface. The numerical abstract domains deal with variables in the mathematical sense, whereas a computer's memory is a sequence of bits, and the same bits can be accessed in several different ways according to the width and type of the value written to/from memory. This distinction cannot be ignored when the target is the analysis of C programs that contain casts. The (non-relational) existing abstract interpretation plug-in of Frama-C will be adapted so as to produce, along with its other results, a mapping from slices of bits to mathematical variables, enabling Apron's relational abstract domains to be used on C programs. This mapping can be produced either "on the go" or so as to be used in a second pass. Both approaches have pros and cons, and some experimentations may be needed in order to decide which is best for the purpose at hand. Lastly, it should be possible in the same framework to introduce variables to Apron that do not correspond to any slice of bits in memory but to a more sophisticated quantity. For instance, introducing variables that correspond to the length of a string would allow to discover buffer overflows automatically, in the fashion of Penjili (<http://www.penjili.org>) or of Moy and Marché early experiments<sup>5</sup> during the CAT project. Other choices of quantities to represent as variables would lead to other interesting properties, which motivates a generic interface where Frama-C's users could decide of arbitrary quantities to represent as variables in Apron.

- Risk Analysis and Mitigation Plan

There is no definitive risk of failure for these tasks as they are incremental improvements and combinations of existing techniques. The challenging issue is the industrial usability of prototypes combining these complex notions. In the worst case industrial partners may wish to limit the possible combinations in order to keep the verification process tractable.

#### 1.5.5 Work Package 4: Proof Management

- Participants: AIF(8), ATOS(12), CEA(10), CNAM(7), CS(6), DA (2), HISP(3.2), **PROVAL(21)**
- Goal

One of the major showstoppers for large programs is that thousands of proof obligations might be generated. Such an amount of proof obligations needs to be handled using an efficient and powerful proof management system. Moreover whenever automated proof fails, the system has to help the user to understand what failed exactly in term of verification. Therefore traceability between proof obligation, source code and formal specification will be implemented together with a tentative counter-example generation. For advanced users, an interactive proof system for first order logic will be defined. This work package has an indirect impact on WP1, WP2 and WP3 in the sense that it will help discharging proof obligations created by the analyzers developed in these work packages. Moreover, it will have a direct impact on WP6 (Spreading Static Analysis), since it will tremendously reduce the time and the amount of human work needed to complete the formal verification of properties of large programs.

---

<sup>5</sup> Inferring Local (Non-)Aliasing and Strings for Memory Safety, HAV 2007, Yannick Moy and Claude Marché

- Detailed Work Plan

The proof management system will offer at least the following three features:

1. Proof Memoisation: A mechanism for storing the current state of proofs for all proof obligations of a given project (annotated source code) will be devised. This mechanism will allow to recover state of proofs between several sessions, to avoid redoing several times the same proof. Moreover, when source code or annotations are modified, a mechanism for merging old proof obligations with new ones will allow to reuse proofs as much as possible.
2. A mechanism of back trace and counterexample generation: This mechanism will help the user understanding proof failures. For a given proof obligation, this mechanism will display an execution path in the source code, together with a set of counterexample values.
3. An interactive proof system: This system will have the charge of handling proof obligations that are not discharged automatically. In such a case, this system will allow the user to interactively select proof tactics to guide automated theorem provers. Such tactics will include *e.g.* hypothesis selection, lemma instantiation, or case analysis.
4. Assessment: Solutions obtained in this WP will be experimented on some representative examples. In particular, some difficult proof obligations encountered during previous evaluations will be used as test cases for understanding proof failure mechanisms and interactive proof tactic definitions.

- Methods and Technical Solutions

Item 1 first requires the design of a format to store proof obligations together with their state of proof. This format will be generic so that it can be handled by several tools and user interfaces. An XML description will be chosen. The merging mechanism may involve several levels of complexity. A simple solution is to perform exact matching on proof obligations, but it might be not powerful enough in practice. More complex solutions should involve semantic criteria such as inclusion of hypotheses.

Item 2 requires handling counterexamples generated by automated theorem provers. This covers two main aspects. First, using our own prover (Alt-Ergo, <http://alt-ergo.lri.fr>), generate counterexamples in a suitable format. Second, external theorem provers, such as Simplify, require some technical work to interpret counterexamples in term of source code.

Item 3 may be achieved using either an existing proof assistant, such as Coq, or by designing a new interactive theorem prover. Implementing a new theorem prover might seem a waste of time but has some advantages. Such a prover can be designed from the beginning with a tight integration with automated provers in mind. Proof obligations can also be displayed as close to source code as possible. Finally, proof tactics can then be designed to fit industrial application needs.

- Risk Analysis and Mitigation Plan

The merging mechanism of Item 1 is challenging. Though, its failure is not critical, since it could be simply implemented using exact matching. Success of Item 2 is related to the quality of counterexamples provided by theorem provers. At least, the back trace mechanism should provide enough feedback to understand the proof obligation meaning. Item 3 potentially requires a large amount of work. To minimize the risk, we plan to provide the possibility of calling another proof assistant, such as Coq.

#### 1.5.6 Work Package 5: Trusting Formal Methods

- Participants: AIF(4), ATOS(5), CEA(20), CNAM(33), CS(4), DA(5), GALLIUM (24), LANDE(30), HISP(3.2), PROVAL(2)

- Goal

Static analysis techniques give strong theoretical assurance that some properties hold on critical embedded software. Implementing these techniques is very difficult and error-prone. This work package aims at increasing the level of trust in concrete static analyzers. More generally, it will investigate how the use of formal methods for developing the tools may impact validation and verification activities. On a methodological point of view, a first task will focus on the “qualifiability” of tools in the sense of safety norms like DO-178B and the forthcoming DO-178C. The other items in this work package will examine the formalization of the most important building blocks of the Frama-C framework into a proof assistant such as Coq. Properties to be proved on these formal models will be given as a result of the first item. A first area of interest is to establish an explicit formal link between the model built by the analyzers (from the source code) and the model used by the compilation tool chain during the translation from source to object code. Second, a certified relational lattice similar to the ones used in task 3.4 will be implemented. Last, we will investigate the feasibility of building a formal model of Scade programs so as to be able to validate the C code generated from Scade (*i.e.* to prove that the semantics of the program is preserved through code generation). As a summary, this work package depends on the models developed in WP3 (Combining Static Analysis Techniques) and, for the Scade models, on the ability to reason about temporal properties, as investigated in WP2 (Temporal Properties). On the other hand, these elements of formalization will foster the usage of the Frama-C platform –the main objective of WP6, Spreading Static Analysis Techniques– in particular in contexts where a very high level of confidence is required.

- Detailed Work Plan

1. Needs and feasibility of qualification of the Frama-C framework tools. We will extract from the software certification processes (in particular those defined according to the international civil avionics norm DO-178B) a qualification strategy. Such a strategy heavily depends on the kind of code, properties and tools selected to perform formal verification. Moreover all prototype developments will take care of keeping the appropriate level of traceability to fulfil the needs of a potential qualification.
2. Linking memory models. Memory models are key elements in the trust one can have in an analyzer. Moreover each analyzer has its own memory model. We will describe the link that must exist between these memory models. Refinement relations will be proved formally. In particular the memory model used by CompCert will be used. Proving that the other memory models are consistent with this model will increase the

trust in verification tool chain, at least when the compiler used is the CompCert compiler.

3. Certification of static analyzers results: Static analyzers are themselves complex programs with technical symbolic manipulations. Abstract interpretation gives a powerful theoretical framework to prove soundness of static analyses with respect to the semantics of the considered programming language. However, a full paper proof is tedious and error-prone when studying real programming languages. This item is dedicated to formally prove within the Coq proof assistant the soundness of the results of a relational static analyser. The analyzer will be developed as a Frama-C plug-in using relational abstraction of numerical variables. The plug-in will be based on the Apron library that gives access to several relational abstract domains with various trade-off between efficiency and precision. On the other side, we will develop a “static analyser checker” in Coq, which will be able to check the result of the analyzer. This checker will be formally proved correct in Coq, with respect to the C semantics developed during the CompCert project.
4. Scade Translation Validation: not all C code produced during embedded software development is written by hand. An important part is generated from higher-level languages and in particular Scade. In this context, it is important to be able to trace back the Scade constructs that lead to a given C code. The aim of this task will be two-fold. On the one hand, the Frama-C framework will provide means to reflect at the Scade level the alarms triggered by the static analyses on the C code itself. On the other hand, we will investigate the possibility of extracting specifications from Scade, in order to prove that the code generation fulfils these specifications. The extensions of ACSL devised in WP 2 will be used in order to build this specification. This task is an experimental study of the more general problem of semantic equivalence between typical high level specification and C code.

- Methods and Technical Solutions

Item 1 is mainly a methodological work that will be led by the industrial partners who have experience in certification activities. The other items require first to define a formal model of the domain under analysis in a proof assistant. Then, the properties of interest of these domains will also be formalized. These properties will be discussed with the industrial partners in order to reflect their certification needs as expressed in Item 1. Last formal proofs of these properties will be obtained. More specifically, item 2 will formally define the various memory models upon which the static analyzers developed in the project are based. Once this has been done, it will attempt to establish the relations that exist between them. Item 3 will focus on the formalization of abstract interpretation lattices in order to provide strong formal guarantees on Frama-C’s results. Last, item 4 is concerned with the formalization of the semantics of Scade programs and the equivalence between Scade and the translated C code. During this task, industrial partners will also express their potential needs related to the validation of both the reflection of C alarms in Scade models and the translation of Scade specifications into C/ACSL.

- Risk Analysis and Mitigation Plan

Given the strong experience of the industrial partners in this field, Item 1 does not present any major risk. The other items are very prospective, especially Item 4, as formal proof development is a challenging activity. All academic partners have a fairly good knowledge of Coq, and will be

able to tackle the issues that may arise during the formalization. Other work packages do not depend critically on these tasks. Item 2 depend partially on Task 3.3, but can base its work on existing memory models if the new ones are not ready as the task starts. Similarly, Item 3 will take as a base the existing Apron library and will not be impacted by the work in Task 3.4.

#### 1.5.7 Work Package 6: Spreading Static Analysis

- Participants: AIF(8), ATOS(18), **CEA(12)**, CNAM(1), CS(8), DA (5), LANDE(6), HISP(3.2), PROVAL(1)
- Goal: The goal of work package 6 is to ensure the long-term success of the project by:
  - Disseminating information about the project, its objectives, the approaches followed and their results.
  - Promoting the use of tools and applications resulting from the project among industrial end-users and within the academic community,
  - Promoting the use of Frama-C as an open framework to implement static analysers,
  - Facilitating the integration of Frama-C in development tools.

Altogether the main goal is to support the user and developer community around Frama-C and static analysis. The results of this work package can be seen as the main success criterion of the U3CAT project as a whole. In this sense, WP6 is strongly connected to all the other work packages of the project, whose developments will be packaged in a single, efficient and easy-to-use tool during WP6.

- Detailed Work Plan
  1. Promotion: In order to enhance the visibility of Frama-C, we will improve the Frama-C website (<http://www.frama-c.cea.fr>) and provide there regular information about the progress of the project. Similarly, a graphic identity in accordance with the principles of the project will strike imaginations and provide the recognition which will ease corporate adoption of the platform. This role will also be played by communication toolkits targeting specific audiences, such as software developers or managers.
  2. Technical Communication: Public talks on Frama-C and ACSL in industrial and academic congresses, articles. Many items in the work packages above contain important scientific challenges, whose issues must be detailed through numerous academic publications.
  3. Technical Support to user and developer communities: This includes an extensive documentation (manuals, tutorials...) and training materials. The Frama-C website will also include convenient feedback possibilities, in particular under the form of a Bug Tracking System (BTS) and a mailing list. Inside the consortium itself, training sessions will be organised.
  4. Extensible Integrated Proof Environment: the existing Frama-C architecture allows to implement separate analyses as plug-ins, which communicate with each other *via* a

relatively small kernel and can present their results through a common user interface. While fully functional, this architecture needs to be refined in order to improve the ergonomics of the framework, both for users and for developers. Work in this item will thus be twofold. On the one hand, the user interface will be designed so as to facilitate aggregating and interpreting the results of different analyses over the same code. On the other hand development and integration of a new analysis in the framework will be made easier in order to lower the level of experience required to develop a Frama-C plug-in. The integration of the tools developed in the other work packages will serve as a guideline for this item.

5. Integration of Frama-C in the Software Development Cycle: this item will investigate how the Frama-C framework could be used in existing development processes. A methodological study will be conducted to understand the impact its deployment would have on software development and certification. Another point of interest here is to figure out how to take advantage of the work in the preceding item to obtain a Frama-C version perfectly tailored for a particular development process.
6. Integration of Frama-C in development tools: Similarly to item 5, we will insert Frama-C in the development process; we will investigate the way the Frama-C tools could be integrated *concretely* within workbenches and tools. This could be done by adapting tools like Topcased (<http://www.topcased.org>) and Papyrus (<http://www.papyrusuml.org>). As a result, we could evaluate benefits of the use of verification done by Frama-C in a more integrated development environment.
7. Assessment on case studies: industrial and academics partners involved into this WP will experiment the main results obtained on code examples representative of realistic applications. One of the major WP success criteria will be the ability to convince different communities (users, developers) about the handiness and efficiency of the proposed solutions.

- Methods and Technical Solutions

The aim of this WP is essentially based on realizing communication means, developing a methodology of use, and validating the overall goals of the U3CAT project through some simple but demonstrative case studies.

- Risk Analysis and Mitigation Plan

Item 1, 2 and 3 do not present any particular risk. Item 4, 5 and 6 are very challenging. Their success is the main criterion for evaluating the whole project. Item 7 will give the final conclusions of U3CAT.

### **1.6. Résultats escomptés et Retombées attendues. Expected results and potential impact. (1 à 2 pages maximum)**

The main expected result of the project as a whole is a significant improvement of the Frama-C framework efficiency and usability for real-life embedded C code verification tasks. This will be evaluated primarily by the industrial partners through the test cases provided in WP 1 and 2. Another expected outcome is to spread the usage of the framework beyond the consortium and to gather a static analysis developers' community around Frama-C. Success will be measured here by the existence of Frama-C plug-ins developed outside of the academic partners of U3CAT. Last, several

scientific challenges are attacked within U3CAT. The success of the project in this area will be measured by the quantity and the quality of the academic publications on Frama-C written during U3CAT.

More precisely, U3CAT expected results can be sorted according to its main domains of interest as they are mentioned in Section 1.3. First, with respect to the enhancement objectives, Frama-C should be able at the end of the project to handle accurately floating point computations and some kinds of temporal properties. This will take the form of new analyses within Frama-C and the integration of existing external tools into the framework. Moreover, the ACSL design document will be enhanced, in particular by providing libraries to ease writing specifications in these new areas. The quality of U3CAT outcome in these domains will be measured by the precision of the analyses over the case studies provided in Tasks 1.1 and 2.1.

When it comes to hardening the existing techniques, U3CAT is expected to offer new ways of combining static analysis techniques, both at theoretical level and as implementation within the Frama-C framework. This should result in significant gains in the precision of the value analysis plug-ins of Frama-C and in the feasibility of analysing code through deductive methods. The ability to address low-level code through weakest precondition calculus is utterly relevant in this respect. Again, the resulting implementations will be qualitatively evaluated on industrial examples.

Another major point related to hardening the current techniques is the proof management system proposed in WP 4. In this area, the main goal is to substantially reduce the time needed to perform formal verification of large applications. This is in particular the case for the proof memoisation tool. As for the traceability features and the interactive proof system, their usability in an industrial environment will be assessed thanks to a thorough interaction between developers and industrial partners.

The third main activity domain of U3CAT is related to the formal verification of some of its fundamental blocks and ultimately to its use in a formal certification context. In this area, technical results can be directly measured by the number of properties that can be formally established about Frama-C inside a proof assistant such as Coq. On a more qualitative level, this formalization effort is meant to ease the acceptance of Frama-C by a certification authority, and in the longer run to lighten the verification effort needed by critical embedded software projects while maintaining safety requirements at their highest level.

As a summary, the expected results detailed above aim at turning the Frama-C framework into an high-quality integrated toolkit for the analysis and verification of critical C code. That point alone would be sufficient to deem the project successful. The existence of a living users' community outside of U3CAT consortium would indicate that this result has been attained. A way to quantify this result would be the number of monthly downloads of the Frama-C platform and the activity of the Frama-C mailing list. However, Frama-C also aims at being an open framework, in the sense that the addition of new plug-ins should require as little effort as possible. Indeed, it should be possible to design very specific analyses, *e.g.* tied to a particular development process. The CAT project has given the basic building blocks that allow such extensions to take place. During U3CAT, it is expected that this blocks will gain their full maturity to provide a fully extensible platform. The creation of a Frama-C developers' community, assessed by the existence of plug-ins developed outside of the U3CAT consortium would be a clear sign that this objective has been fulfilled.

**1.7. Organisation du projet. Project management : structure and flow.**

**1.7.1 Relations between the Work Packages.**

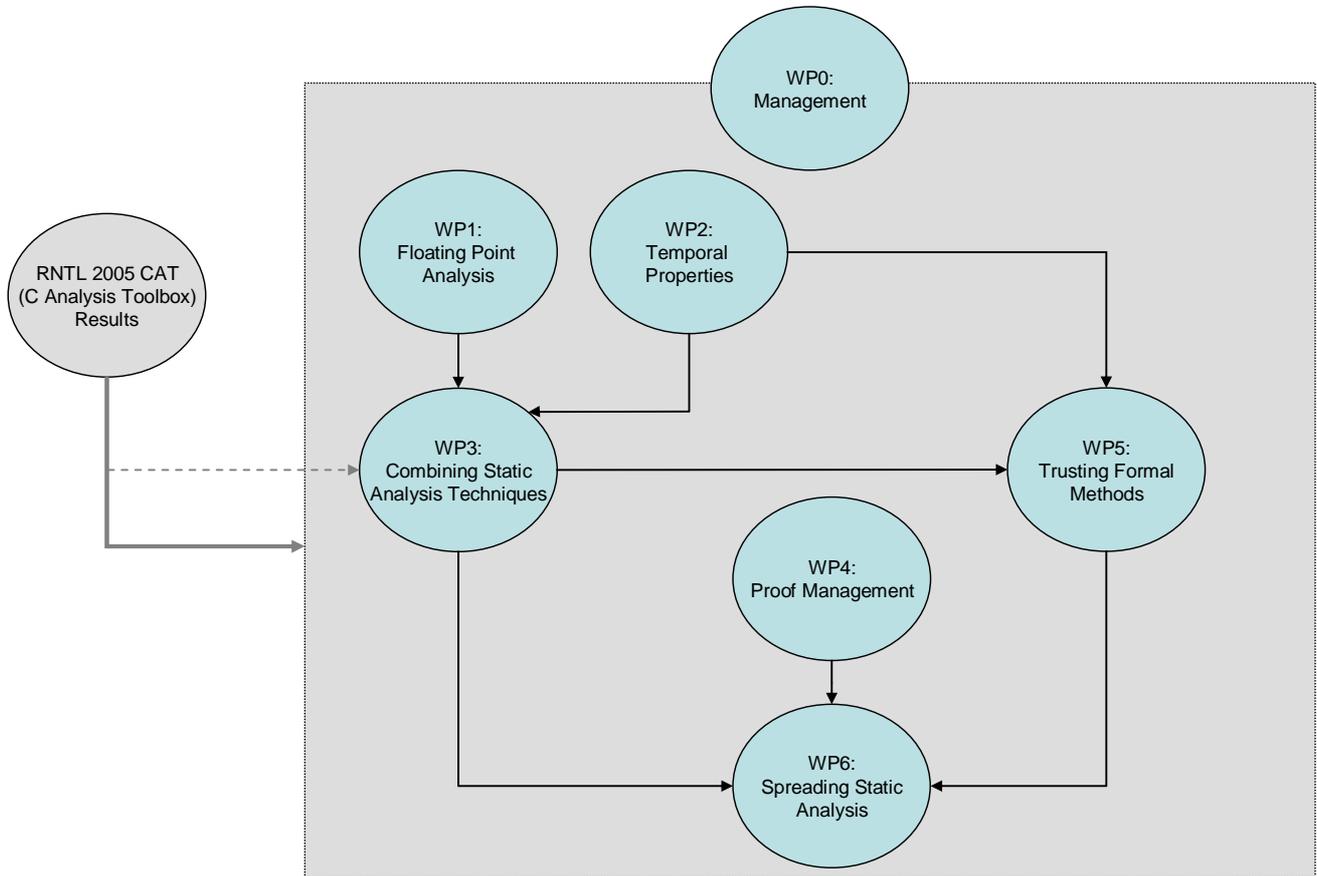


Figure 1: Relations between the Work Packages

U3CAT project will enlarge and enforce RNTL 2005 CAT results, as it will combine (in WP3) previous tools and methods obtained in CAT project with Floating-Point (WP1) and Temporal Property approaches (WP2). Then, together with an efficient Proof Management system (WP4) and improving the confidence in Formal Method results (WP5), partners of the project will have at their disposal a global approach of Static Analysis to be integrated in their own industrial processes and/or to be disseminated over C Critical Code industrial communities (WP6).

	Chronogramme / chemin critique ( <i>Timing diagram/ critical path</i> )		
Partenaires/ <i>Partners</i>	Année / <i>Year 1</i>	Année / <i>Year 2</i>	Année / <i>Year 3</i>

	C E A	P R O V A L	G A L L I U M	D A	C N A M	L A N D E	A I F	A T O S	H I S P	C S	2	4	6	8	10	12	16	18	24	34	36		
Task 0	■										■	■	■			■			■		■	■	
Task 1	■	■		■		■		■			■	■	■	■	■	■	■	■					
Task 2	■					■		■			■	■							■				
Task 3	■	■			■			■					■	■	■	■	■	■					
Task 4		■				■							■	■	■	■	■	■					
Task 5					■														■		■	■	
Task 6	■										■	■	■	■	■	■	■	■	■	■	■	■	
Livrables /Jalons																				M 1		M 2	
Deliverables/Milestones																							
Rapports d'avancement / états des dépenses												☺			☺			☺			☺		☺
Progress report/expenses												☺			☺			☺			☺		☺
Accord de consortium/rapport final																☹							☹
Consortium agreement/final report																☹							☹

- ☺ : 6 month-progress report
- ☺ : Progress report + expenses
- ☹ : Consortium agreement
- ☆ : Final report + expenses summary
- : Work package leader
- : Work package participant

M1: Milestone 1, Tool prototypes covering new domains and hardening static analysis techniques for industrial use

M2: Milestone 2, Integrated Proof Environment with trust elements

**Note:** the timing diagram indicates the most active periods for each work package, *i.e.* the period when the formal deliverables must be provided. In particular highly prospective items may start ahead of this schedule and last up to the end of the project.

The table below summarizes the deliverables for each Work Package in the project. Reports are indicated by an 'R', software by an 'S', other kinds of deliverables by an 'O'. The milestones of the project can be identified according to the three main research domains detailed in Section 1.3:

1. Tool prototypes related to new domains and to hardening the existing framework will be delivered at T0+24
2. Tool prototypes integrating trust elements and reports on the formalization of the techniques will be delivered at T0+36

In addition to the formal deliverables listed below, new versions of the Frama-C platform will be released on a quarterly basis. Each release will include the analyses that have gained a sufficient level of maturity to be used outside of the project. Release cycles will be conducted following the “Cathedral” model<sup>6</sup>, *i.e.* they will be led by the CEA.

No formal deliverable is planned for Tasks 6.1 (Web site animation) and 6.2 (Communication). The web site will be updated continuously (in particular, but not solely, for the project’s milestones), so that there is little point in fixing particular delivery dates. Talks and articles related to U3CAT will be reported in the semestrial reports. Similarly, not all activity in Task 6.3 (User and Developer Support) will give rise to a formal deliverable.

<b>TABLEAU des LIVRABLES et des JALONS (le cas échéant)/ Deliverables and milestones</b>			
<i>Tâche/ Task</i>	Intitulé et nature des livrables et des jalons/ Title and substance of the deliverables and milestones	Date de fourniture à compter de T0/ Delivery date, in months starting from T0	Partenaire responsable du livrable/jalon / Partner in charge of the deliverable/ milestone
<b>0. Management</b>			
	O – Kick-Off Meeting	T0	CEA
	R – Semestrial Report 1	T0+6	CEA
	R – Semestrial Report 2	T0+12	CEA
	O – Project Consortium Agreement	T0+12	CEA
	R – Semestrial Report 3	T0+18	CEA
	R – Semestrial Report 4	T0+24	CEA
	R – Semestrial Report 5	T0+30	CEA
	R – Final Report	T0+36	CEA
<b>1. Floating Point Analysis</b>			
	R – Initial Specification (Item 1)	T0+6	PROVAL

<sup>6</sup> as in Eric S. Raymond, *The Cathedral and the Bazaar*, O’Reilly, 2001.

	S – Abstract Interpretation Lattice for Floats (Item 2)	T0+16	CEA
	R – Feasible path analysis over floating-point computations (Item 2)	T0+16	LANDE
	S – Memory Models for Floats (Item 3)	T0+12	PROVAL
	R – Evaluation of the Tools	T0+18	AIF
<b>2. Temporal Properties</b>			
	R – Initial Specification (Item 1)	T0+8	CEA
	R – ACSL Extension (Item 2)	T0+10	PROVAL
	S – Adaptation of Existing Techniques (Item 3)	T0+14	CEA
	S – Application of Model Checking Techniques (Item 4)	T0+18	CEA
	S – Predicate Abstraction (Item 5)	T0+22	PROVAL
	R – Evaluation of the Tools	T0+24	DA
<b>3. Combining Static Analysis Techniques</b>			
	S – Combination of Abstract Interpretation and Weakest Pre-condition (Item 1)	T0+14	PROVAL
	S – Combination of Slicing and Weakest Pre-condition (Item 2)	T0+12	CEA
	S – Combination of Memory Models (Item 3)	T0+17	PROVAL
	S – Combination of Abstract Interpretation Lattices (Item 4)	T0+17	CEA
	R – Evaluation of the Tools	T0+18	CS
<b>4. Proof Management</b>			
	S – Proof Memoisation (Item 1)	T0+17	PROVAL
	S – Counterexample Mechanism (Item 2)	T0+18	PROVAL
	S – Interactive Proof System (Item 3)	T0+23	CEA
	R – Evaluation of the Tools	T0+24	AIF
<b>5. Trusting Formal Methods</b>			
	R – Qualification Strategy for Frama-C Tools (Item 1)	T0+26	AIF
	R – Memory Models Formalization (Item 2)	T0+34	GALLIUM

	R – Certified Lattices (Item 3)	T0+34	LANDE
	SR – Scade Translation Validation and Industrial Assessment (Item 4)	T0+34	CEA
6. Spreading Static Analysis			
	R – Updated User and Developer Documentation (Item 3)	T0+30	CEA
	S – Integrated Proof Environment (Item 4)	T0+34	CEA
	R – Integration in Software Development Process (Item 5)	T0+30	ATOS
	S – Integration in development Tools (Item 6)	T0+36	ATOS

## 1.8. Organisation du partenariat. Description of the Consortium.

### 1.8.1 Pertinence des partenaires. Presentation of the relevance of each partner to the proposal.

#### Airbus France [AIF]

The interest of Airbus France for static analysis techniques started in 1996 with the LAW IST project, which contributed to the development of a research prototype based on Hoare logic: Caveat (CEA). Afterwards, the interest of Airbus also focused on the application of the Abstract Interpretation theory to software verification. The company's involvement in this area started with its participation in DAEDALUS (IST 5th PCRD) as the unique end-user of the project. Airbus' constant objective in this field is the transfer of software verification techniques from fundamental research to Airbus development teams by means of static analysers. Examples of such tools already used for certification credit in avionics software development are: Caveat (CEA), aiT and Stackanalyser (both from AbsInt GmbH). Next candidates for transfer include Astrée (ENS), Fluctuat (CEA), and Frama-C.

One of Airbus' primary objectives in the CAT project was to extend the scope of the proof of functional properties beyond the limitations of the existing Caveat tool: increase the rate of automatic proofs, deal with aliases, casts, floating-point numbers and temporal properties. The former will be met by the end of the CAT project, whereas U3CAT is definitely needed to meet the latter. Airbus will thus express needs about such features, and assess tools on real-world embedded software.

Another interesting feature of Frama-C arose during the CAT project. Its modular structure should make it possible for advanced users to develop their own specialised analysis plug-ins, in order to automate the verification of specific design or coding rules. Airbus is willing to investigate this possibility.

Finally, Airbus is involved in the development of international aeronautics regulations (DO-178B and forthcoming DO-178C), which should help addressing certification and qualification issues.

#### ATOS Origin [ATOS]

Atos Origin has a longstanding experience in critical and non-critical software development in the C language and has gathered a solid knowledge in the domain of formal proof since 2002. In this context, Atos Origin has a practical user experience in an industrial context of activities like formal

#### U3CAT

specification and interactive proof. It has developed and/or integrated verification-tools with the qualification constraints of the DO178-B and in relation with research partners. In addition, as a member of the committee in charge of the definition of the DO-178C, Atos Origin has a theoretical and practical background in the certification process.

Atos Origin is confident in the advantages of these techniques and is interested both in applying them internally (increased productivity and quality during development processes, impact analysis on TMA projects) and in promoting them to industrial end-users.

For those reasons, Atos Origin will put efforts on extending the application domain of these techniques with temporal properties (task 2), facilitating qualification of the tools (task 5), facilitating the use of them in particular condition such as manual proof termination (task 4) and, at last, providing methodological guidelines in line with existing development tools (task6).

#### **CEA List – Software Reliability Laboratory [CEA]**

The Software Reliability Laboratory (LSL) has an outstanding record track in program analysis techniques. Its static analysis team has developed an expertise in abstract interpretation and Hoare logic, as well with formal methods and the use of proof assistants. It has successfully developed and maintained the CAVEAT tool, which has been used at Airbus during the certification process of some part of the A380 embedded software. More recently, the LSL has been heavily involved in the development of the Frama-C framework, notably for the Frama-C kernel itself, the abstract interpretation plug-in and the slicing plug-in.

#### **CNAM Cedric laboratory [CNAM]**

The CPR team of CEDRIC research laboratory is interested in formal methods applied to the development of confident software. His members share an important knowledge in general systems like B and Coq as well as in semantics and type systems. CPR participates in the development of the CompCert certified compiler since its beginning in 2003.

In the context of the CompCert project, CPR has contributed mainly to the development of the memory model of the compiler, as well as adequate formal semantics for C-like languages and also separation logic for the Cminor intermediate language, that is a first bridge between program proof and the compiler verification effort

#### **CS Communication & Systèmes [CS]**

The dual position of CS, as a multi-domain integrator of critical embedded systems and as a service provider, independent of any particular tool vendor, allows CS to offer innovative verification processes particularly well fitted for critical systems, even in the case of processes deeply modifying existing ones. U3CAT techniques and technologies belong to these areas of innovative verification means for embedded systems: CS will then efficiently contribute to the project, on one hand providing relevant industrial use cases, on the other hand giving its feedback and experience in method and processes improvements and modifications. CS will be essentially involved in evaluations in Tasks 3 and 4, and methodological aspects of work package 6.

The future DO-178C certification standard will probably be available close to the launch of the U3CAT project. CS is involved in the definition of this new issue of the Aeronautical Software Certification standard. Therefore, contributing to answer the question: “how to be compliant with this new standard using formal techniques” will be a supplementary added-value that CS will bring in the

U3CAT

frame of certification aspects of work package 5, and methodological aspects of work package 6. Connected to that question, some internal research activities aim at introducing model checking techniques within an existing Model Driven Engineering industrial process. The results of this work are strongly linked with work package 6. Finally, the strong involvement in the ITEA ES\_PASS project, is another illustration of interest of CS high performance verification techniques like abstract interpretation.

#### **Dassault Aviation [DA]**

In the last ten years, Dassault Aviation has actively participated to several research collaborations (from which RNTL 2005 CAT) and projects (as Usine Logicielle in the French competitive cluster System@tic) with academics (CEA, LRI, INRIA, ...), or advanced tool assessment on industrial use cases, in the domain of formal verification for safety properties on its critical embedded applications. These works dealt with formal specification (control-flow and data-flow) and model-checking, imprecision error analysis, proof of program and theorem proving, on large size or algorithmically complex C source code.

In U3CAT, Dassault Aviation will participate in expressing needs and lessons learned from past experiences, and assessing whole methods and tools. For these assessment issues, Dassault Aviation will identify, extract - and obfuscate if needed for dissemination purposes - some realistic use cases. Methodological aspects will be an important part of the whole assessment of the project solutions, as it will enforce the coherence of the different approaches U3CAT deals with.

All significant results during the assessments will permit to consider the integration of the innovative methods and tools into its software development platforms, the objective being to make these different approaches contribute to certification process of its embedded critical C codes.

#### **Hispano-Suiza [HISP]**

For more than ten years, formal verification with model-checking has been assessed on significant pilot projects in the frame of the European IST Safeair projects where Snecma then HISPANO-SUIZA were the coordinator. This technique is used on several projects to check some properties and perform equivalence between Stateflow and corresponding C code from the SCADE environment. More recently, HISPANO-SUIZA investigated tools based on abstract interpretation in the frame of the ModRiVal sub-project of Usine Logicielle in the French competitive cluster System@tic with academics (CEA, LRI, ...) and purchased tools for Worst Case Execution Time and Stack analysis.

Hispano-Suiza has published (with CEA LIST/LMeASI) on experiments made with the static analyzer FLUCTUAT on control programs, which is of direct interest to the current project. Hispano-Suiza's rôle will be mostly to give relevant industrial code, assess the relevance of the choices made during the project, and disseminate the results obtained in industry. Dissemination will be done in the SAFRAN Group through the SEED (Safran Embedded Engineering Development) where HISPANO-SUIZA belong to as well as Messier-Bugatti, Sagem Defence Security and Turbomeca. Inside SEED, HISPANO-SUIZA has a leading role for Static Analysis. Dissemination will also be achieved externally in the aeronautics industry by contributing to conference and workshops in cooperation with the academics.

All significant results during the assessments will allow to consider the integration of the innovative methods and tools into its software development platform, the objective being to make the U3CAT toolset contribute to certification process of its embedded critical C codes.

#### **U3CAT**

Finally, as Airbus, HISPANO-SUIZA (previously Snecma Control Systems and Elecma in the 90's) is involved in the development of international aeronautics regulations (DO-178B and forthcoming DO-178C), which should help addressing certification and qualification issues.

#### **INRIA project-team Gallium [GALLIUM]**

The Gallium project-team of INRIA Paris-Rocquencourt works on the design, implementation, formalization and formal verification of programming languages and systems, including the popular Objective Caml functional language. In the context of the CompCert project, Gallium developed and proved correct, using the Coq proof assistant, a realistic optimizing compiler for a large subset of the C language. Gallium brings to UC3AT the expertise thus gained on mechanized operational semantics, axiomatic semantics and memory models for C-like low-level languages, as well as their uses to reason over compiler passes and other program analyses and transformations.

#### **INRIA Project-team LANDE [LANDE]**

The INRIA's LANDE project-team of IRISA is concerned with formal methods for verifying and validating software. The LANDE project-team's foundational activities are concerned with the semantics-based analysis of the behaviour of a given program. These activities draw on techniques from static and dynamic program analysis, testing and automated theorem proving. In terms of static program analysis, our foundational studies concern the specification of analyses by inference systems, the classification of analyses with respect to precision using abstract interpretation and reachability analysis for software specified as a term rewriting system. Particular analyses such as pointer analysis and control flow analysis for C have been developed. For the implementation of these and other analyses, we are improving and analysing existing iterative techniques based on constraint-solving and rewriting of tree automata. Concerning static analysis, we have in particular investigated how flow analysis of programs based on constraint solving can help in the process of generating test cases from programs and from specifications. More speculatively, a long-term goal is to integrate the techniques of abstraction and constraints into a common framework for approximating program behaviour. LANDE has also developed an experience in the formal certification of abstract interpreter inside the Coq proof assistant. Several certified static analyses have been developed for bytecode Java, using the abstract interpretation theory: data-flow analyses<sup>7</sup>, resource usage analysis<sup>8</sup>, and interval analysis<sup>9</sup>. Among the tools developed and maintained by the project, \*FPSE\*<sup>10</sup> is a symbolic evaluation tool for C programs and CLL (Certified Lattice Library)<sup>11</sup> is a library of Coq modules to build certified analyses.

The LANDE participants to the UC3AT project are currently involved in the European Integrated Project Mobius (2005-2009) dedicated to Proof Carrying Code for bytecode Java programs and the SESUR 2007 CAVERN project (2008-2011) dedicated to the combination of abstractions and constraints for verifying rule-based and imperative programs. As said previously, existing results of the CAT project will serve as inputs to the UC3AT project but there is no overlapping as the main

---

<sup>7</sup> Cachera Jensen Pichardie Rusu ESOP 04

<sup>8</sup> Cachera Jensen Pichardie Schneider FM 05

<sup>9</sup> Besson Jensen Pichardie TCS 06

<sup>10</sup> Gotlieb Botella Watel ICSSEA 06, Botella Gotlieb Michel STVR 2006

<sup>11</sup> Pichardie FICS 08

theme of the implication of LANDE in this new project is on floating-point computations and certified static analysis.

The group involved in U3CAT consists in David Pichardie (CR INRIA), Arnaud Gotlieb (CR INRIA) and Frédéric Besson (CR INRIA).

#### **INRIA Project-team PROVAL [PROVAL]**

The Proval team (<http://proval.lri.fr/>) works on static program verification, focusing to deductive methods based either on automated or interactive theorem provers. They develop the Why platform for deductive verification of both C and Java code. This platform has been developed and distributed for several years (<http://why.lri.fr/>), and has been used for several industrial case studies: Java Card applet verification in collaboration with Gemalto<sup>12,13</sup>, verification of C code in collaboration with Dassault Aviation<sup>14</sup>. ProVal already contributes to the Frama-C platform, inside the current CAT ANR project. They quite recently started working on floating point number issues in verification<sup>15</sup> and on combination of abstract interpretation and weakest precondition calculi<sup>16</sup>. On the theorem proving side, they develop their own automatic prover Alt-Ergo (<http://alt-ergo.lri.fr/>) and contribute to the Coq proof assistant. The team also quite recently hired Marc Pouzet who works on synchronous programming, with applications to the SCADE compiler.

#### **1.8.2 Complémentarité des partenaires. Description of complementarity within the consortium.**

The consortium gathers several industrial partners representing industrial domains where embedded software plays a very important role (automotive, aeronautics and spatial industries), and widely recognized academic partners which have a strong scientific background in the domains investigated by the project. All of the members of the consortium have already collaborated with at least one of the other partners of the project. In particular, the majority of the participants have taken part to the CAT project, where the collaboration between industrial and academic partners has been proved very fruitful in order to foster research and development efforts toward the issues encountered in real-world certification process. This new consortium shows the will of the partners to pursue this collaboration on the longer run. In particular, the presence of IT service companies in the consortium presents an important advantage for a wider adoption of static analysis techniques for software verification.

---

<sup>12</sup> Bart Jacobs, Claude Marché, and Nicole Rauch. Formal verification of a commercial smart card applet with multiple tools. In AMAST'04, volume 3116 of Lecture Notes in Computer Science

<sup>13</sup> Claude Marché and Nicolas Rousset. Verification of Java Card applets behavior with respect to transactions and card tears. In SEFM'06, IEEE Comp. Soc. Press

<sup>14</sup> Thierry Hubert and Claude Marché. Separation analysis for deductive verification. In HAV'07.

<sup>15</sup> Sylvie Boldo and Jean-Christophe Filliâtre. Formal Verification of Floating-Point Programs. In 18th IEEE International Symposium on Computer Arithmetic

<sup>16</sup> Yannick Moy. Sufficient preconditions for modular assertion checking. In VMCAI'08, volume 4905 of Lecture Notes in Computer Science.

### **1.8.3 Qualification du coordinateur du projet. *Principal investigator: skills and CV.***

Benjamin Monate is a former student of the École Normale Supérieure de Cachan (97). He got a computer science PhD in 2002 at the Paris XI University of Orsay under the supervision of Jean-Pierre Jouannaud and Evelyne Contejean about parameterized string rewriting. He spent year 2003 at the LIX computer science laboratory as “Chargé d’Enseignement de l’Ecole Polytechnique”. Since 2004 he is the scientific leader of the static analysis activities in the LSL (Software Security Laboratory). He has a strong management experience in national (RNTL 2005 CAT (leader), DGE PFC ...) and European (ITEA €-confidential, ITEA ES\_PASS, IP OpenTc...) research projects and in direct contracts with the industry.

### **1.9. Stratégie de valorisation et de protection des résultats. Data management, data sharing, intellectual property strategy, and exploitation of project results. (1 page maximum)**

The code developed or acquired before the start of the project will remain the property of the concerned partner. This is in particular the case for the industrial case studies, which will be made available to the other relevant partners to evaluate the tools developed in U3CAT, but won’t be disclosed outside of the consortium without the explicit consent of their respective owners. Each partner will have the property of the tools that it will develop during the project. All tools will be freely available for all partners for the work conducted in the context of the U3CAT project. Developments that are deemed critical for the Frama-C kernel will be released under an Open Source license, compatible with the LGPL 2.1 under which the Frama-C framework is currently available. For more specific plug-ins, the choice of releasing the code or not, and in the former case, of a proprietary or Open Source license will be left to its owner as far as the project dissemination objective is not impaired. Project management will ensure that the points above will be formalized through a consortium agreement signed by all partners in due time.

The choices of licenses for each part of the existing Frama-C software platform (LGPL 2.1 and the BSD license) have been made in order to allow the emergence of an ecosystem around it.

Proprietary plug-ins can be developed to interface with it, and service, methodology and expertise can be provided by private interests.

These same choices also make the platform sustainable. There is no single point of failure that would make the platform undependable. As a large public technological institute, one of the primary goals of the CEA is to maintain the products of successful projects until they are ready for a full industrial transfer. Thus, the CEA will provide an appropriate level of support for the Frama-C platform on the longer run beyond U3CAT. The development of the Caveat tool is a previous example of such an involvement of the CEA regarding its industrial user Airbus France. This strong commitment from the CEA and the “Cathedral” model of development chosen for the Frama-C platform ensure that changes in the consortium involved in the Frama-C development (members entering or leaving) will not endanger the continuance of the platform itself.

## **2. Justification scientifique des moyens demandés. *Requested budget : detailed financial plan.***

### **2.1. AIRBUS France.**

#### **2.1.1 Large equipment.**

N/A

#### **2.1.2 Personnel.**

Permanent: 36 person.month, *i.e.* 1 engineer full time for 3 years (305 118 €)

#### **2.1.3 Services, outward facilities.**

N/A

#### **2.1.4 Travels.**

Projects meetings: 1 person, 3 times a year for 3 years (5 400 €)

#### **2.1.5 Expenses for inward billing.**

N/A

#### **2.1.6 Other expenses.**

N/A

### **2.2. ATOS Origin**

#### **2.2.1 Large equipment.**

N/A

#### **2.2.2 Personnel.**

Permanent staff (engineer): 40p.month

#### **2.2.3 Services, outward facilities.**

N/A

#### **2.2.4 Travels.**

Congresses and project meetings: 1 person, 3 times a year, during 3 years, *i.e.* 5778€

#### **2.2.5 Expenses for inward billing.**

N/A

**2.2.6 Other expenses.**

One desktop workstation and one laptop: 3210€

**2.3. CEA List**

**2.3.1 Large equipment.**

N/A

**2.3.2 Personnel.**

Permanent staff: 110 person.month, *i.e.* a little more than 3 persons per year full time (803 247 €)

A lot of prototypes are to be developed by the CEA and necessitate experienced researchers and developers to reach the level of maturity expected by industrial partners.

**2.3.3 Services, outward facilities.**

N/A

**2.3.4 Travels.**

Congresses and project meetings: 3 persons, 3 times a year, during 3 years, *i.e.* 13500€

**2.3.5 Expenses for inward billing.**

N/A

**2.3.6 Other expenses.**

3 personal workstations or laptops: 6000€

**2.4. CNAM-CEDRIC laboratory.**

**2.4.1 Large equipment.**

N/A

**2.4.2 Personnel.**

Permanent staff: 19 person.month *i.e.* 1/3 of the full time “Maître de Conférences” Sandrine Blazy and 1/5 of the full time “Professeur d’Université” Catherine Dubois during 3 years.

Engineer: 24 person.month *i.e.* 2 years of a full time junior engineer (6 p.m on WP4 and 18 p.m on WP5)

**2.4.3 Services, outward facilities.**

N/A

**2.4.4 Travels.**

Project meetings, international and national congresses: 5000 € per year during 3 years, *i.e.* 15000 €

**2.4.5 Expenses for inward billing.**

N/A

**2.4.6 Other expenses.**

One desktop workstation and one laptop: 3000 €

**2.5. CS**

**2.5.1 Large equipment.**

N/A

**2.5.2 Personnel.**

Permanent staff: 24 person.month consisting in experimenting the tools in WP3 and WP4 (12 p.m), contributing to WP5, Trusting Formal Methods, (4 p.m), and participating to the integration of Framac in software development processes (WP6, 8 p.m).

Total: 332160€

**2.5.3 Services, outward facilities.**

N/A

**2.5.4 Travels.**

Participation to congresses and project meetings: 1 person during 3 years, 3 times a year: 4500€

**2.5.5 Expenses for inward billing.**

N/A

**2.5.6 Other expenses.**

1 personal workstation or laptop: 2000€

**2.6. Dassault Aviation**

**2.6.1 Large equipment.**

N/A

**2.6.2 Personnel.**

Permanent staff: 24 person.month, consisting in specifying industrial needs, assessing academics tools and participating to dissemination of U3CAT results. Moreover, Dassault Aviation's involvement into technical discussions with academic partners will contribute to technology transfers from Research labs to industry R&T teams.

**2.6.3 Services, outward facilities.**

N/A

**2.6.4 Travels.**

1200 €: Participation to congresses (for dissemination purposes) and U3CAT meetings

**2.6.5 Expenses for inward billing.**

N/A

**2.6.6 Other expenses.**

N/A

**2.7. Hispano-Suiza**

**2.7.1 Large equipment.**

N/A

**2.7.2 Personnel.**

20 person.month consisting in participation to all WPs with the centre of gravity in WP4 assessments, and dissemination activities.

**2.7.3 Services, outward facilities.**

N/A

**2.7.4 Travels.**

6000 €: participation to project meetings, and conferences, two in Europe and two in France.

**2.7.5 Expenses for inward billing.**

N/A

**2.7.6 Other expenses.**

16324 € covering the costs for providing computing facilities to the principal investigators of the project.

**2.8. INRIA-Gallium project.**

**2.8.1 Large equipment.**

N/A

**2.8.2 Personnel.**

Permanent staff: 12 person.month *i.e.* 1/3rd of the full time DR2 INRIA Xavier Leroy during 3 years.

Post-Doctoral or Expert Engineer: 12 person.month *i.e.* 42000 €

**2.8.3 Services, outward facilities.**

N/A

U3CAT

**2.8.4 Travels.**

Project meetings, international and national congresses: 5000 € per year during 3 years, *i.e.* 15000 €

**2.8.5 Expenses for inward billing.**

N/A

**2.8.6 Other expenses.**

One desktop workstation and one laptop: 3000 €

**2.9. INRIA-LANDE project.**

**2.9.1 Large equipment.**

N/A

**2.9.2 Personnel.**

Permanent staff:

- David Pichardie (CR2 INRIA) 12 p.month, that is 33% for the 3 years on WP 5.3 and WP 6.2.
- Arnaud Gotlieb (CR1 INRIA) 12 p.month, that is 33% for the 3 years on WP 1.1 and WP 6.2.
- Frédéric Besson (CR1 INRIA) 9 p.month, that is 25% for the 3 years on WP 5.3.

Non-permanent staff:

- One 18 months post-doctoral position on WP 1.1.
- One 12 months internship position on WP 5.3.

**2.9.3 Services, outward facilities.**

N/A

**2.9.4 Travels.**

Participation of principal investigators and the post-doctoral researcher to the two annual meetings: 16 national trips, *i.e.* 4 000 € (16 x 250€)

Participation to three international conferences for the principal investigators and the post-doctoral researcher each year during the project: 8 international trips), *i.e.* 12 000€ (8 x 1500€)

Total : 16 000 €

**2.9.5 Expenses for inward billing.**

N/A

**2.9.6 Other expenses.**

Two laptop computers for the temporary staff (one post-doc and summer interns) hired on the project.

Total: 4000€

## **2.10. INRIA-Proval project.**

### **2.10.1 Large equipment.**

A server dedicated to computation, with several processors (8) and a large amount of memory (16 GB). Its main use is related to WP 4 on Proof Management, but it will also serve to any case study developed in the project. Such a multi-processor system will allow to run several provers on several verification conditions in parallel. We ask for 7500 € for that purpose.

### **2.10.2 Personnel.**

Permanent staff:

- Claude Marché, DR2 INRIA, 14 p.month, that is 39% for 3 years
- Jean-Christophe Filliâtre, CR1 CNRS, 12 p.month, that is 33% for 3 years

Non-permanent staff: Post-Doctoral: 24 p.month, 90602 euros. 12 p.m in WP2 and 12 p.m on WP4

### **2.10.3 Services, outward facilities.**

N/A

### **2.10.4 Travels.**

Congresses and project meetings: 3 persons, during 3 years, 20000 €

### **2.10.5 Expenses for inward billing.**

N/A

### **2.10.6 Other expenses.**

3 personal workstations or laptops: 6000 €

## **Annexes**

Description des partenaires/ Partners informations (cf. § 1.8.1)

### **AIRBUS France**

Airbus is a leading aircraft manufacturer that consistently captures around half of all orders for airliners with more than 100 seats. Their product line-up, which covers a full spectrum of four aircraft families from a 100-seat single-aisle to the largest civil airliner ever, the double-deck A380, defines the scope of their core business. Their mission is to provide the aircraft best suited to the market's needs and to support these aircraft with the highest quality of service.

In 2006 Airbus achieved a turnover of around 26 billion euros and provided support for the highly reliable operation of the more than 4, 500 Airbus aircraft currently in operation with more than 270

U3CAT

operators around the globe. To date, Airbus has received more than 7,000 orders from some 250 customers.

Based in Toulouse, France, Airbus is an EADS company, incorporated under French law as a simplified joint stock company or "S.A.S." (Société par Actions Simplifiée). It employs directly some 57,000 people of over 80 nationalities. Airbus' design and production sites are grouped into four wholly-owned subsidiaries, Airbus France, Airbus Deutschland, Airbus España and Airbus UK.

Airbus France designs and manufactures aircrafts, aircraft parts, systems, equipment and derivative products, and also provides services in the field of aeronautics. About 15,600 employees are working in Airbus France.

The team involved in U3CAT works within a department of Airbus France where avionics software is been developed for the most critical systems of Airbus aircrafts. This team aims at transferring static analysis tools from research to operational development teams.

### **ATOS Origin**

Atos Origin is a leading international IT services provider. We provide integrated design, build and operate solutions to large multi-national clients in carefully targeted industry sectors. Our business approach is based on establishing long-term partnerships that encourage success through mutual benefit.

### **CEA-LIST**

The CEA is a Government Research Institute employing 15000 persons. It is a major player in research, development and innovation. CEA intervenes in three main fields: energy, information and health technologies and Defence.

The LIST is a CEA laboratory of applied research on software-intensive technologies mainly on embedded systems, interactive systems and signal detection and processing. With its 450 researchers, the LIST performs research works in partnership with numerous university laboratories as well as with the major industrial players in nuclear, automotive, aeronautical, defence and medical fields.

The Software Reliability Laboratory (LSL) team works on tools for the verification and validation of software and hardware systems (in particular system on chip). It has built up, and continues to develop, an expertise in some of the most advanced techniques in fields like: static analysis using Hoare logic, abstract interpretation, rewriting and proof techniques, simulation, constraint logic programming, and source code instrumentation.

### **CNAM-CEDRIC**

The CNAM is a Public Scientific, Cultural and Professional Institution, classed as a *grand établissement*, among France's top higher education establishments. Supervised by the Minister for Higher Education, the Cnam has a triple role: providing life-long training; undertaking research in technology and innovation; disseminating a culture of science and technology.

The CEDRIC research laboratory is the biggest laboratory inside CNAM. It brings together researchers from CNAM and ENSIIE. Most of them are computer scientists, the others are mathematicians. The CEDRIC research topics cover mainly five themes: development of reliable

### **U3CAT**

systems, combinatorial optimization, information system and databases, networks, system and multimedia and statistical methods for data-mining and learning.

The CPR (Conception et Programmation Raisonnées) is interested in formal methods applied to the development of confident software. The main subjects of interest concern the development of specification and proof languages, formal semantics aided by proof assistants, rewriting and certification of automatic proofs. In particular, CPR is involved in the development of the FOCAL integrated development environment, the rewriting tool Cime3, the CERPAN project and the CompCert certified compiler.

## **CS Communication & Systèmes**

Designer, integrator and operator of mission critical systems, CS defines itself both as a systems integrator and as an IT service company. This exceptional positioning enables CS to offer its customers in France, Europe and around the world, with operations in defence, aeronautics, space, energy, transportation, the public sector, and finance:

- implementation of large, complex, high-value projects,
- integration and a comprehensive vision – Design/Build/Run,
- innovation in high performance and technology.

Centers of Technological Excellence as a prime contractor and operator on large, complex projects, CS relies on its centers of technological excellence in order to meet the high-stakes challenges of its customers:

- simulation and virtual reality,
- embedded systems,
- communication and information system security,
- safety and continuity of operations,
- technical information systems (PLM, EDM, GIS, and others).

CS centers of technological excellence enable it to deliver innovative, turnkey solutions to its customers, and to ensure performance and continuity for their systems.

- 10 patents and 10 software programs registered in 2006
- A network of 200 specialists and experts
- Participation in 5 world-renowned competitiveness clusters.

CS contributes to large French and European R&D programs.(e.g. the FP6 ASSERT project recently completed) It is also very active in various French competitiveness clusters, responsible among other tasks for the coordination of the ES-PASS project (Embedded Systems Product bases ASSurance , an ITEA project focused on critical software verification with static analysis techniques) in the Aerospace Valley cluster, and the IOLS project (Infrastructures et Outils Logiciels pour la Simulation, a national project on infrastructures and software tools for simulation) of the System@tic cluster.

## **Dassault Aviation**

U3CAT

Dassault Aviation is one of the major players in the global civil and military aviation industry, a reasonably sized and financially secure private international group, with a presence in more than 70 countries across 5 continents. Structured to adapt its production to market cycles, Dassault Aviation encompasses a rich industrial network of high-tech companies in France, Europe, the US and many countries worldwide.

Through its R&T and engineering design departments, production facilities, the skills of its employees and its product lines, Dassault Aviation offers its customers in-depth know-how, ranging from design to operations, based on strong entrepreneurial values. Technological excellence and innovation is the motto of Dassault Aviation on which its spirit, passion and history are based. The Group ensures the quality, reliability, and safety of its aircraft through a strategy of constant innovation, its project management capability, and its mastery of complex systems.

Research and development is essential to the Group's activity in terms of preparing for the future.

The upstream understanding of innovative technologies that are the most promising in terms of cost/efficiency, through fundamental research, has always been considered a primary factor in competitiveness.

The general research conducted by Dassault Aviation enables the application of new technologies to both current programs and future systems. Particular attention is given to the work which reduces program cycles and costs and improves quality and safety. The Company conducts studies and research under internally financed projects and in partnership with the French government and European institutions. It maintains partnerships with over one hundred research centers in France and worldwide: universities, laboratories, institutes, manufacturers, etc.

### **Hispano-Suiza (SAFRAN Group)**

<http://www.hispano-suiza-sa.com/>

Hispano-Suiza, a SAFRAN Group company, is a world leader in mechanical power transmissions, electronic engine controls and control systems for airplane and helicopter engines. Hispano-Suiza has delivered products for more than 21,000 commercial and military engines to date, logging nearly 390 million hours in flight. Drawing on this technical and industrial expertise, Hispano-Suiza is already developing new control and power management solutions for the next generation of aircraft engines.

Hispano-Suiza is a pioneer in power electronics for severe environments, and is working on the technologies and systems required for tomorrow's "more electric" aircraft. Hispano-Suiza delivers total solutions to aircraft engine manufacturers, to optimize engine operation under actual conditions.

Hispano-Suiza products account for about 30% of the total value of a military aircraft engine, and about 20% on their civil counterparts. Whether control, power transmission or other functions, Hispano-Suiza designs the system in close collaboration with the engine-maker - that's our job as a systems integrator-engineering firm. Our overriding objective is of course the continual improvement of all criteria that will ensure the engine operates at maximum efficiency. Hispano-Suiza then draws up specifications and produces equipment, which is integrated in the engine.

The team involved in U3CAT works within the Electric Division where avionics software is been developed for each and every critical systems. This team aims at assessing and transferring formal verification tools from research to operational development teams.

### **U3CAT**

## INRIA

INRIA, the national institute for research in computer science and control, operating under the dual authority of the Ministry of Research and the Ministry of Industry, is dedicated to fundamental and applied research in information and communication science and technology. It hosts 2900 research scientists in its eight research centres. INRIA's research is organized around seven priority grand challenges, one of which is particularly relevant to this proposal: guaranteeing the reliability and security of software-intensive systems.

Three INRIA project-teams participate to the UC3AT proposal: Proval (Saclay research centre), Lande (Rennes research centre) and Gallium (Paris-Rocquencourt research centre).

The **Gallium** project-team works on programming languages, type systems, compilers, and their formal verification using the Coq proof assistant. Its flagship projects include the Objective Caml functional programming language and system, and the CompCert verified C compiler for critical embedded systems.

The **Lande** project-team works on static analysis and software testing, targeting mobile code such as Java bytecode and embedded programs such as critical C programs. The focus is on providing methods with a solid formal basis (in the form of a precise semantics for the programming language used and a formal logic for specifying properties of programs) in order to provide firm guarantees as to their correctness. The group involved in U3CAT consists in David Pichardie (CR INRIA), Arnaud Gotlieb (CR INRIA) and Frédéric Besson (CR INRIA). Some of them are currently involved in the SESUR 2007 CAVERN project (2008-2011) and the European FET/IP "Mobius".

The **Proval** project-team works on static program verification, focusing to deductive methods based either on automated or interactive theorem provers. They develop the Why platform for deductive verification of both C and Java code, and contribute to the Frama-C platform. On the theorem proving side, they develop their own automatic prover Alt-Ergo and contribute to the Coq proof assistant. Finally, a fairly recent axis of research is on synchronous programming, with applications to the SCADE compiler.