Proofs of Functionality for Mobile Distributed Systems

PROFUNDIS

Proposal No: IST-2001-33100

September 17, 2001

Updated April 10, 2002

Contents

1 Project summary

1.1 Objectives

The objective of PROFUNDIS is to advance the state of the art of formal modelling and verification techniques to the point where key issues in mobile distributed systems, such as security protocols, authentication, access rights and resource management can be treated rigorously and with considerable automatic support. In particular we shall verify properties typical in so called open systems, where the behaviour of some parts (like intruders or adversaries) is unknowable, in extensible systems, where parts may be added or removed as the system executes, and in mobile systems where physical and logical connectivity between parts may change. We shall implement automatic and partly automatic analysis methods for ascertaining correct behaviour of such systems. For this purpose we shall integrate and focus several strands of ongoing theoretical work.

1.2 Description of Work

The work builds on recent advances in key theories for process behaviours, logics and types. We shall develop automata theoretic models suitable for our applications, with a particular interest in how they can be represented efficiently and used by automatic tools, and we shall determine how they are best used in connection with advanced forms of modal logics. The logics themselves will be developed, both in terms of their expressiveness for properties related to space and structure, and in terms of their accessibility and ease of use through suitable high-level representations.

We shall identify and develop analysis techniques related to these models and logics. This involves traditional behavioural equivalences and preorder checking, systematic simulation, and verification in interactive proof assistants. Here type systems will play an important role. Recent results show that types may themselves be used as crude but tractable correctness properties and therefore type inference is highly relevant, moreover, we shall explore how advanced type information can assist the other analysis techniques.

The ideas will to a large extent be implemented in a common tool set. Key issues here will be development and adaption of algorithms for analysis, and determining the best way of using them for practical examples. We shall in particular consider examples on security properties in systems for electronic commerce.

1.3 Expected Results

The results from the project will be of two kinds. First there will be a set of tools that automatically or with some manual assistance can treat the verification problems in the objectives above. The tools will be accompanied with a pragmatics, i.e., a guide for how they are effective for particular problems.

Second the theories underpinning the tools will be demonstrated in scientific publications.

2 Project objectives

The objective of PROFUNDIS is to advance the state of the art of formal modelling and verification techniques to the point where key issues in mobile distributed systems, such as security protocols, authentication, access rights and resource management can be treated rigorously and with considerable automatic support. In particular we shall verify properties typical of: (i) systems with incomplete information about the environment, where the behavior of some parts (like intruders or adversaries) is unknowable; (ii) systems where parts may be added or removed as the system executes; and (iii) mobile systems, where physical and logical connectivity between parts may change. We shall design and implement automatic and partly automatic analysis methods for ascertaining correct behavior of such systems. For this purpose we shall integrate and focus several strands of ongoing theoretical work.

One strand represents mobile distributed systems as variants of automata for the purpose of automatic verification. Here we shall in particular focus on so called HD-automata. In them, processes differing for permutations of free names are represented by the same state. HD automata are able to allocate and garbage collect names and can be minimized and model checked; they are finite for finite control agents; they provide an intermediate, syntax independent format and they have been used for verifying π-calculus processes via translation to ordinary automata. Since translation is possible only in restricted cases, a key objective here is to develop an algorithm for direct HD-automata minimization and to implement it in an automatic tool.

The need of accommodating more general kinds of calculi suggests to extend HD-automata in several ways. This will be accomplished with more general notions of name substitutions, like non-injective renamings (so called fusions) and cryptographic primitives. We shall also include suitable forms of parallel composition, boxing and name binding operators, which can be used to express spatial structures. The resulting intermediate language will be clause-based, will include simple data structures and will allow to embed many modern calculi (like π-calculus, spi-calculus, join calculus, ambient calculus etc). We shall then determine which properties are of practical interests, are decidable and possess proof techniques which are efficient enough. The techniques we shall devise are automatic whenever possible (model checking, bisimilarity checking) and semiautomatic otherwise (symbolic execution, coinduction proofs). In particular advanced techniques will be used to express security properties as semantic equivalence.

Independently from HD-automata, but in relation with them whenever possible, we will develop a logical framework to support the specification and verification of correctness properties addressing both structure and behavior of concurrent mobile systems. The framework shall accommodate properties in-

volving handling of names, security and secrecy features, location-dependent access rights to resources, location dependent-behavior, invariants on the dynamic communication topology and routing, dynamically created objects and references. To support formal reasoning an adequate level of abstraction, new logics shall be developed incorporating specific constructs to express higher level properties of space and structure, and their interaction with time and behavior. By jointly addressing structure and behavior of processes, it will be possible to tackle the richer computational phenomena associated with mobility, coordination, and dynamic software architecture.

We shall devise proof systems for the logic and determine its expressiveness. The higher-level extensions will particularly suit our applications, allowing specifications in a more concise and comprehensible—hence usable—way. An important objective is to build tools for semiautomatic verification which are both usable in practice and general enough to cope adequately with the difficult system features which will be central to the highly mobile and dynamic applications envisaged in the future.

Related to both models and logics we shall develop type systems. We shall accomplish two main kinds of results. Firstly, the types themselves express properties of processes. In this sense they support the logic and enable concise and accessible formulations of correctness properties. Here, our emphasis will be on how types allow us to control interferences among processes and their accesses to system resources. The second kind of results is the use of type information throughout automatic verification, like minimization and model checking, in order to increase efficiency. For instance, certain types guarantee that certain communications are not preemptive. This is a partial confluence property, in the presence of which only parts of process behaviors need be explored.

The type systems so far studied can only handle rather basic properties, and have been developed in isolation and on small process calculi. Our aim is to arrive at a general and flexible framework where several type systems can be accommodated and which can be applied to practical languages and systems. Examples of properties that are important in practice and that we wish to handle using type systems are: resource consumptions (time or memory allocations needed by a process to carry out a task); declassification, i.e., the ability to downgrade some confidential information temporarily (this feature turns out to be essential when dealing with applications for e-commerce); secure composition of components. Another important objective is the integration of types with the techniques based on automata and logics. This will require an accurate comparison of the associated proof techniques and of the expressiveness of different formalisms (in particular those based on logics vs those based on types), and refinements of the standard notion of observation on which the theory of 'classical' process calculi (CSP,CCS, π-calculus,etc.) is based.

To show the practicality of the type systems, we will study the automatization of the proof techniques based on types. This will include type-inference algorithms that verify whether a given process is type-correct, which require as little as possible information from the user.

3 Participant list

R = Role, N = Partner number, Cy = Country

R	N	Participant name	Short	Cy	Entry	Exit
C	1	Uppsala University	UU	S	start of project	end of project
P	2	Fundação da Faculdade de Ciências e Tecnologia	FFCT	P	start of project	end of project
P	3	Institut National de Recherche en Informatique et en Automatique	INRIA	F	start of project	end of project
P	4	Università di Pisa	Pisa	I	start of project	end of project

4 Contribution to programme/key action objectives

PROFUNDIS aims at making global computing systems more trustworthy by providing tools and methods for catching errors in their designs before they are implemented and employed. In this sense the project directly addresses one of the fundamental concerns of IST : "... the development of an efficient networking and computing infrastructure together with advanced mobile and networked embedded systems that enable anywhere/any time access to services. This requires new tools and business models for service design ..." (All quotations here are from the IST Workprogramme.) Methods and tools of the kind we want to provide have been proven useful in other specialized areas such as hardware construction or protocol analysis, but they are still elusive for the complexities of the systems targeted by the Global Computing call: systems that are "... rapidly growing and changing computational environments composed of highly diverse interconnected devices. In these environments, everything is dynamic: physical devices are mobile, connectivity and bandwidth are changing and computational processes and data migrate." The challenge is to treat these systems on a sufficiently high level of abstraction so that analysis techniques are tractable. If successful, our results will have a profound impact on the way systems are designed and to which degree they can be trusted and consequently can gain acceptance.

This requires a substantial development of underlying theories. The important aspects are "... trust and security, including information security, privacy, ... and dependability of systems and infrastructures...", one of the priorities within IST. Clearly, the effort in PROFUNDIS is by nature high-risk, and good enough conditions for such an ambitious project does not exist at any single

site; the effects on the market will be long range but potentially very large. Moreover it will be difficult to directly patent or protect knowledge of the basic nature that we seek. For these reasons we are in compliance with the IST policy line which encourages projects "Anticipating market needs and nurturing emerging technologies where public funding can make a substantial impact, by aggregating fragmented research and building critical mass ahead of market maturity."

The Global Computing call identifies three key aspects. The first, "The design of systems composed of autonomous entities whose participation in the computation is dynamic and where activity is not centrally co-ordinated" is directly at the heart of PROFUNDIS; we shall treat exactly such systems and focus on design based on formal methods. The second is "Analyzing and reasoning about the behavior of such systems, both qualitative and quantitative, even when very large numbers of entities or interactions are involved." Within this aspect our project shall focus on the qualitative aspects of early and high-level designs of systems. The third is "Avoiding and/or detecting undesirable behavior through control of the system and/or its environment". In fact we aim at a methods where it is possible to rigorously state what constitutes desirable or undesirable behavior, something which is quite difficult with the formal models that exist today.

5 Innovation

Innovation in PROFUNDIS will be of two kinds. First there will be a development of the existing theories for specification and verification of mobile, dynamically changing systems with incomplete information about the environment, extending their applicability and integrating several theories in common frameworks. Second there will be a development of automatic and semi-automatic tools, together with a collection of case studies and pragmatics demonstrating how the tools can be used efficiently. The innovation here lies in the range of applications that can be treated by the tools.

Existing calculi for specifying distributed, interactive, mobile systems share a common emphasis on name creation and handling. Also names play an important role in practical network applications. Nonces identifying sessions in security protocols and dynamic events of network environments are instances of the same phenomenon: fresh name generation. Also free names can be adequately used to identify (links to) unknown agents and attackers. Following some recent advances in the area of higher order abstract syntax [GP99, PG00], partner PISA has recently studied models called HD-automata [59, 57] which can be characterized as coalgebras in a category of name permutation algebras. Semantically equivalent states, or agents, are mapped into the same state of the final coalgebra. This framework is sufficient to capture the notions of free name, name passing and name generation that are needed to describe and reason about dynamically changing systems. It has been shown that synchronous

and asynchronous π-calculus agents with their various semantics (early, late, open) can be mapped into (finite or infinite) HD-automata. A related approach based on indexed labelled transition systems has been proposed by [CS00].

Finite state automata (e.g. finite labelled transition systems) provide a basic model underlying effective verification techniques of concurrent systems. Efficient algorithms and techniques have been developed to automatize the verification process for model checking and semantic equivalence, and they are essentially the only formal verification methods widely used in practice. Finite state techniques have been used also to analyze cryptographic protocols [Low96, CJM98, MMS97]. However, to keep the state space finite, these tools make simplifying assumptions in the model. For instance, they assume a bound on the size of the messages synthesized by the adversary. Also, with the exception of the tools [15, 51] developed by partners UU and PISA , usually only straight-line protocol - or cyclic protocols executed a small number of times - are analyzed, which allows to replace freshly generated names by fixed constants. In many practical cases these assumptions are unrealistic, and thus the applicability of finite state techniques is severely limited.

Within PROFUNDIS we shall develop innovative finite state representation techniques together with expressive logical frameworks and effective verification techniques to certify features of mobile systems. With certification we mean the ability of formally expressing and verifying properties of network applications (e.g. security levels of applications, trust in migrating components, safe and authorized resource accesses). HD-automata provide a promising approach, since they often succeed in abstracting out the irrelevant name identities and in providing a small finite state model. By now HD-automata have been used as an intermediate language to represent finite control π-agents, but they have actually been translated to ordinary automata for verification. Since translation is possible only in restricted cases, a key objective of the project is to develop an algorithm for direct HD-automata minimization and proof checking and to implement it in an automatic tool

Symbolic execution may effectively improve the analysis process by constraining the way a partially known component can contribute to overall systems behavior. In protocol analysis, symbolic execution allows automatic verification without simplifying assumptions in the formal model [24, 22, 23, 39]; indeed desirable properties such as secrecy or authenticity are specified by inserting logical assertions in the code describing the protocol. Again, HD-automata are likely to allow for a symbolic execution mode which could make at least straight-line protocols verifiable.

We shall extend the theory of HD-automata in several ways to accommodate a variety of calculi. This will be accomplished with more general notions of name substitutions, like non-injective renamings (so called fusions) and cryptographic primitives. We shall also include suitable forms of parallel composition, boxing and name binding operators, which can be used to express spatial structures. The resulting intermediate language will be clause-based, will include simple data structures and will allow to embed many modern calculi (like π-calculus, spi-calculus, join calculus, ambient calculus, etc). We shall then

determine which properties are of practical interests, are decidable and possess proof techniques which are efficient enough.

Bisimilarity is one of the most popular techniques for defining behavioral equivalence on processes. With respect to other notions of process equivalence, the main advantage of bisimilarity is the associated co-inductive proof method, whose applicability and usability can be greatly enhanced by means of the so-called "up to" technique. Proving processes equivalent has been shown remarkably useful for verifying many security properties [FGM00]. The idea is to suitably modify the system by exposing the possible effects of some attack: if the original and the modified system are semantically equivalent, the system has not been damaged by the attacker.

Logics for concurrency and mobility have been studied extensively [HM85, 11, 3]. Most proposals are based on the modal mu-calculus. Sophisticated model-checking techniques and tools have been developed by partners UU and PISA for the automatic verification of finite-state systems e.g. [15, 51]. Other approaches [27, RE99] have explored the application of theorem proving techniques to the theory of bisimulations, successfully handling some classes of infinite-state systems. Compositional proof techniques relying on sequent calculi have also been developed, and applied to process terms [4] and even to programs written in industrial strength high-level languages [1].

Innovation in PROFUNDIS includes development of logics with sufficient expressive power to characterize not only the time-structure of labelled transition systems, but also the space-structure of each state. Recently, some logics have been proposed by Cardelli and Gordon [CH00, CG01] and by partner FFCT [18, 17] with the motivation to capture notions of spatiality and behavior of processes. These logics are of quite a low level and do not directly express many of the properties we want to cope with. We thus shall introduce other, more useful, higher-level structural operations and associated specialized proof principles. The challenge is to obtain a logic framework which can handle rich features like exceptions, links, resources, nonces, names, dynamically created object and references, locations and administrative domains.

We shall introduce several alternative notions of "space". For example, spatial logics may take meaning over a forest of trees or graphs. Then, a forest of trees is well suited to represent hierarchies of administration domains inside wide area networks. Labelled graphs provide an alternative basic structure for describing the topology (spatial distribution and connectivity) of networks.

Within PROFUNDIS we shall develop proof systems for these logics based on sequent calculi. We want to consider both proof-systems where entailment is interpreted in terms of the properties expressible by logical formulas relative to a class of models, and proof systems for specific operational models, which will be targeted at proving properties of concrete process terms.

We shall further develop a logical framework to support the specification and verification of properties required for the correctness of the highly mobile and dynamic applications envisaged in the future. Properties that shall be studied

in this setting include security and secrecy, location-dependent access rights to resources, location dependent-behavior, handling of names, constraints on dynamic topology and routing, and even failures. These properties have not yet been properly addressed by existing logics for concurrency and mobility.

The research field on types for mobility is rather young (about 8 years old). Initial work on the area has aimed at transporting well-known type systems for sequential calculi, in particular λ-calculi, to process calculi. More powerful forms of types have then been introduced. For this project, we are interested in types that express controls of interferences, access controls, bounds on resource consumptions (for instance time or memory allocations). We describe below the existing work on types that should be most relevant for the project.

Capability types provide information about the access rights on available resources (channels, sites, etc.). Capability types have been introduced by partner INRIA [30] – following Reynolds's type discipline for local variables [Rey88] – in the simple setting of the standard π-calculus, and have then been extended in a number of ways. Most significantly, partner PISA has extended them to distributed languages for coordination [46], in which types are network-aware. Abadi [Aba97] uses types for guaranteeing secrecy properties in security protocols. The types used by Abadi combine the idea of the capability types with that of types for information-flow. The latter types, which are best known through the works of Smith and Volpano [SmVo98], guarantee that private information is not improperly disclosed. In a collaboration, partners INRIA and PISA [54] have applied ideas of capability types to define threads of control in processes of distributed calculi (Ambient-like).

Another class of type systems for which behavioral consequences have been studied are those based on *linear* (i.e. "use once") typing [Hon93, KPT99, 34]. The basic mechanisms of linearity have more recently been extended in certain cases to type systems capable of guaranteeing properties such as deadlock freedom [Yos96, IgKo01, SuKa98, RaVa97].

Partner INRIA has also used types based on *existential polymorphism* [31] to express information hiding barriers, for instance to hide the implementation details of some resource, as in Mitchell and Plotkin's abstract data types [MiPl88].

A number of researchers, including some at partner INRIA , have used proof techniques based on types to prove the validity of algebraic laws and program transformations on object-oriented languages [29, 32, Dal98, HKMN99].

In process theory, roughly, two processes are equal is they cannot be distinguished by any contexts of the given process language. Such a notion of equivalence is however awkward to use in proofs. Thus one employs stronger versions based on the actions a process can perform. In typed calculi, however, the ordinary notion of action observation is much too strong. The innovation for the project here will be the refinement of the classical notions of observables and actions which can accommodate typed mobile processes. Some work in this direction has been recently done (for instance in a collaboration between partner INRIA and partner PISA [42]), but for very basic type systems and, even

in these ad-hoc cases, the picture is not fully-developed.

We should point out that type systems for interference, access control, and related issues are still a serious challenge also on sequential languages. Research carried outside the project that will be closely monitored include that on Proof Carrying Code [Nec97], Typed Assembly Language [MWCG98], and resource bounds [CrWe00, Hof99].

The main limitations of existing type systems for mobile processes are: types can only express very simple properties; each type system has been developed in isolation and on very basic formalisms (typically dialects of the π-calculus); the type checking rules are complex and not algorithmic.

The innovations of the project will consist of: i) the collection of properties which can be handled with types will be significantly enlarged, considering properties that are challenging but essential in practical applications. Examples are declassification of access rights and memory consumption; ii) the transfer of type systems from foundational calculi to real programming languages. For this, programming constructs such as local variables, exceptions, objects, will be considered. This transfer will allow us to test our type systems on realistic case studies; iii) the integration of different type systems into a single general and flexible framework; iv) the development of algorithms for carrying out the analysis of type systems mechanically. As elsewhere in the project, we will focus on techniques for automated analysis.

6 Community added value and contribution to EU policies.

The last decade has witnessed an enormous development in the field of mobile distributed computing, and the situation is still evolving. As wireless networks become commonplace, all kinds of portable devices will be linked to fully connected computers anytime and wherever one goes. Numerous platforms for various forms of mobile code have recently appeared and continue to appear. As the supporting infrastructure becomes more widespread, new applications and services will become available.

Unlike the standalone applications that are still the great majority today, the mobile applications will migrate on networks, as when a user process moves to a server location to execute a task on behalf of its owner. A crucial issue in this kind of systems is to find precise ways to reason about them, to specify their properties and to ensure that they behave as desired. A related issue is that of security, mobile entities being more exposed than standalone entities to typical security attacks like unauthorized access or denial of access. We aim in this project to tackle these issues by integrating different but complementary approaches and techniques, and implementing them into automatic or semi-automatic tools.

The range of problems dealt with in the project calls for diverse competences that can not be found in a single place. Models, logics and types are the different

perspectives that are put together to contribute to the overall goal of developing computationally feasible verification methods. No single approach is expected to solve this problem by itself. On the contrary, it is sensible to explore different approaches, sometimes complementary and sometimes in competition, to assess the strong points and the weaknesses of each, and to integrate them in the design of efficient methodologies and tools.

European universities have a long tradition and a leading position in the fields of calculi for mobility, verification, theorem provers and model checkers. All participants bring their expertise in some of the above fields (which some have pioneered as well), but none could find the complementary expertise at national level. We are confident that the added value of our collaboration in the project will allow us to obtain results beyond what would be achieved on an individual basis.

7 Contribution to Community social objectives.

The field of mobile distributed computing is relatively young but promises to have a strong positive impact in most aspects of our lives. To realize this promise, the field must prove itself in the commercial world, which in turn requires the prior development of appropriate theories, methodologies, techniques and tools, including those of the kind proposed in the project. An ill-founded subject can undermine the confidence of the economic agents and the public alike in the supporting technology, with undesired consequences to its future development and commercial exploitation. Our effort, directed at making mobile open systems more trustworthy, contributes to meet the social objectives of the Community of improving the quality of life and safety.

This project addresses the central issues of computationally feasible verification methodologies and security that proved their value in certain areas of computer science, but whose study is much more difficult and has barely begun for mobile computing. The results obtained, if the project is successful, will contribute to the design of reliable and robust systems with high quality of services and to the development of technologies that enhance trust and confidence. It is not expected that the results will be immediately used by the industry at the end of the project, but we will take steps to train young researchers and to transfer the results to industry. The European industries can gain a crucial competitive advantage by adopting verification technologies that improve the quality of their products. It is foreseeable that start-up companies can be created specifically for that purpose. In the end they will need highly qualified personnel for the task, which will contribute to create new employment opportunities.

New technologies require training new people at the academic level. A positive impact is also expected at the level of graduate and postgraduate teaching in which most of the participants are involved.

8 Economic development and S&T prospects

We expect that the results of the project will have a strong scientific impact, by significantly advancing the state of the art on verification and tools for process mobility. We will strive to disseminate our results by publishing in the best journals and conferences on the topic of verification and mobility. Other dissemination media will include:

- the public WEB site of the project, where general information about the project, public deliverables, and output produced (papers, open software) will be available;
- the participation to the project workshop of a few external persons (say 3 or 4), from academic and industry. This will also contribute to monitor the activities external to the project, and to get useful feedback.

Besides this, all participants in the project are willing to integrate their experience and results in tutorials and courses, with the purpose of training young researchers and promoting the transfer to industry.

The problem of provably-correct software is one of the greatest challenges that underpin the success of Global Computing (security, integrity, quality of service). The theme of the project lays well in this fundamental strategic direction. The problems we have to tackle are hard and, realistically, we cannot promise that by the end of the project the output will be immediately usable by industry.

The goal of the project is rather to develop techniques, methods and, where applicable, prototypes, from which the technological transfer towards the languages and the formalisms in use in industry is realistically feasible. The case studies carried out within the project will serve as demos, with which we hope we can convince industry to undertake the transfer. Depending on how the project evolves, whenever possible other forms of technological transfer may be considered (the Steering Committee will monitor that).

The work of the project has potential implications in many application areas, including: electronic commerce transactions, telecommunications, medical filing, microprocessor cards, electronic forms of authentication, and interactions among these.

Below, we explain how each individual participant intends to disseminate and exploit the results of the consortium.

Uppsala University

Uppsala University is one of the major universities in Sweden. There is a strong emphasis on information technology through formation of a recent large department and new educational programmes. This development is conducted in co-operation with major industries, and UU has a commitment to facilitate new start-up companies as spin-offs from research projects. The infrastructure for this already exists and will be greatly expanded. We foresee two different routes

for dissemination and exploitation, beside the normal scientific publications and conference participation.

One route is to influence education. UU is in a good position with its new information technology programme, which emphasizes the connection between society and technology. The programme will have a considerable amount of freedom in choosing the application areas. It is likely that the results from PROFUNDIS, where successful, can be used in applied student projects, thereby promoting the understanding of importance and techniques in modern formal methods for system design. Project courses currently being taught include tools such as SPIN and the Concurrency Workbench. This demonstrates a readiness to include such aspects in education; through PROFUNDIS, UU will be able to extract state-of-the-art techniques which are more relevant and effective.

Another route is commercialization through start-up companies. Here UU in collaboration with the Royal Institute of Technology and the Swedish Institute for Computer Science has a good track record, and is active in working for collaborations and exchange of ideas in national competence centers. There are several small companies specializing on formal design techniques, making good progress by consulting in sectors of telecommunications, railroads, automobiles, and electrical power systems. It is conceivable that either new start-ups can be formed, or an existing one can adopt new techniques. To this end it is probably important that the results from PROFUNDIS are public; what makes such companies successful are more expertise and efficiency in practical work than possession of IPR for fundamental results.

FFCT

Being a research and teaching institution, the FFCT will exploit the results of the project in improving its teaching (new courses, possibly at the postgraduate level, projects for students) and in creating new research opportunities: deepening the ties with the groups in the project, with other Portuguese groups working in the same subject area, submitting new project proposals on a national or European levels. The FFCT has launched in 1986 Uninova, a research institute that acts as a university/industry interface, and which has helped several start-up companies to come into existence in the area of the information technologies. Mutual exchanges between those companies and the FFCT are encouraged, which creates good conditions for the transfer of the results obtained in the project.

INRIA

As a major research center with strong orientation towards technological transfer, INRIA is well positioned to ensure that the output of the project will be exploited. INRIA is very interested in using the results of the project for enhancing existing collaborations with industry (Gemplus, Trusted Logic) and also for extending the collaborations to other companies. The INRIA group is very interested in studying whether the techniques and knowledge that the

group has produced and acquired in the domain of types and models for mobile processes can be transferred to realistic languages, and then into industry. INRIA also wants to use the project in training and formation activity both at post-graduate and post-doctoral level, and in fostering the collaborations with the other research centres in the project.

PISA

The "Dipartimento di Informatica" of the University of Pisa had a leading role in developing computer science in Italy: it offered the first undergraduate and PhD courses in computer science and in many aspects led research and transfer to industry. Also in Pisa are active IEI and CNUCE, two large institutes of the Italian National Research Council (CNR) which are the Italian counterpart of ERCIM. A strong interest is now present in Pisa for research and applications in the field of wide area network programming, including mobile and nomadic computing, web information retrieval and grid computing, as also shown by the presently active project in collaboration with Microsoft Research and academic groups in Bologna, Florence and Milan. The results of the project will be employed first to foster the research strength of the academic groups, then to transfer the results to several small and medium companies already existing in the area. Also, well established connections with the main Italian industrial research laboratories, like Telecom Italia Lab (formerly CSELT), will be used to increase collaboration on related themes. Also IRST (Instituto per la Ricerca Scientifica e Tecnologica, Trento), where Marco Pistore (which collaborates in the project) is employed, is interested in participating to the workshops, in studying the techniques and tools for certifying security properties that the project will develop and in comparing and integrating them with the technologies it is adopting.

9 Workplan

9.1 General description

The project contains one workpackage for management and three scientific workpackages (WP1-3) representing three different stands of verification techniques, based on automata, logics, and types. This diversity does not mean separation of activities, but it rather reflects a complementarity of competences among the participants. Indeed an important merit and a motivation of the project is to promote interactions among researchers involved in the three strands. We expect that this will produce crossfertilisation of ideas, and transfer and integration of techniques. For more details, see for instance, the theme 'Models for spatial logics' in task 1.1, 'Proving spatial properties' in task 1.2, task 2.1, 'Operational and logical techniques for typed calculi' and 'Space in types' in task 3.2, 'Implementations' in task 3.4.

WP 1 focuses on models of our systems under study. We shall use variants of finite automata, so called HD-automata, and extend them to suit our needs.

We shall devise appropriate verification and proof methodologies with emphasis on automation. WP 2 focuses on logics for expressing specifications of relevant properties to verify. An important point will be to extend traditional modal logics with connectives expressing intensional structures representing administrative domains with explicit access to resources. We shall here devise proof systems and high-level logics for expressing and verifying properties in a convenient way. WP 3 focuses on type systems for mobile open applications, treating issues like process interference, capability types, and secure compositions. Particular emphasis shall also be on integration of type systems and the way types can be used in the formalisms developed in WP 1 and WP 2, and on scalability to realistic programming languages.

Each WP has a task about tools and case studies, where the theoretical results are transferred into working prototypes treating verification examples from areas like security in open mobile systems. The exact structure of our tools will be determined by the success of the WPs in formulating applicable results. Originally there will be one tool in WP1 focusing on automatic analysis and one tool in WP2 on interactive verifications, with WP3 providing input and extensions to both of these. As the project progresses, this structure may be reassessed. For example the tools may merge or new prototypes may prove more fruitful.

As in all projects of this kind, there is a risk of not progressing according to plan because a particular line of research proves too difficult. It is probably not relevant to formulate detailed contingency plans at this stage. But each WP treats a spectrum of problems of varying difficulty, and reassignment of goal priorities depending on the likelihood of success will be made by the involved research leaders and discussed at the workshops.

The effort of each WP is estimated in person-months per participant. These figures refer to to the total effort of PROFUNDIS, i.e., also including personnel at our sites contributing to the project but not directly funded by it. In contrast, the administrative forms and budget report person-months only for people directly funded by the project.

Workpackage 1: Models

The objective of the workpackage is to develop a comprehensive automata-like model and the corresponding verification environment that supports effective and efficient techniques to certify properties of network applications. The model will supports both the handling of names in its full generality and symbolic execution. We also intend to develop efficient and specialized proof techniques and algorithms which help in proving properties.

Task 1.1: Automata with operations and substitutions

The goal of this task is the development of a new automata-like model, a generalization of HD-automata, which supports both the handling of names in its

full generality and symbolic execution. This model will be used to give meaning to a variety of notions for calculi and logics studied within the project.

The task will systematically develop step-by-step all the features which will be needed in the model. We will proceed by isolating and studying interesting fragments of the model and establishing completeness and expressiveness results for them. We also plan to investigate several notions of preorders, equivalences and congruences associated with the models. The structure of this task is further detailed into four themes.

Automata with name fusions This theme aims at extending the permutation model, and correspondingly HD-automata, to handle *identification of names*. This generalization will be sufficient to support in a semantically correct way verification for the fusion calculus [14] (which is a variant of the π-calculus where input and output are symmetric operations).

Automata with operations This theme is concerned with the definition of the algebraic structure of states. Beside permutations and fusions, the algebraic structure should include operations for system composition, like parallel composition, restriction, ambient creation, etc.

Automata with substitutions In this theme we will develop a notion of automata as a coalgebra with general substitutions rather than just renamings (name permutations). The algebraic structure of states will be extended to the general model as well.

Models for spatial logics In this theme we will apply the above automata for the spatial logics investigated in WP2. We intend to investigate classes of applications by exploiting the different classes of models developed in the previous themes. These will be then applied to the corresponding process calculi and logics.

Task 1.2: Proof techniques

This task will focus on the development efficient and specialized proof techniques and algorithms which help in proving properties. The ultimate goal of the task is to contribute to the assessment of the model and to understand the effectiveness and the usability of the associated verification environment (Task 1.3). The structure of this task is further detailed into four themes.

Symbolic execution In this theme we will explore a methodology based on symbolic execution for the automata model developed in Task 1.1. We intend to investigate algorithmic approaches, namely, efficient solutions to the problem of generating symbolic successor states of a given state of the model. We also intend to develop a comprehensive theory and the corresponding verification techniques for the symbolic analysis of security protocols. The symbolic analysis

will exploit the automata with substitutions since it can be regarded as a variant of term unification. Finally, symbolic execution will be employed to visit the state space of open systems.

Verification via proofs of semantic equivalence Many security properties can be verified by modifying the specification of a protocol by exposing the effects of a possible intrusion and by checking if the original protocol and the modified protocol are semantically equivalent. For multiple session protocols ordinary finite state verification techniques (e.g. checking bisimilarity) are not fully adequate. We propose to analyze several classes of security and access right protocols and to verify them by automata minimization.

Due to the rich structure of messages exchanged and to the power of the adversary the reachability problem for cryptographic protocols becomes quickly undecidable. An interesting solution to this problem consists of developing approximated techniques to produce conservative analyses of the protocols. Preliminary results concerning the reachability problem are presented in [Mon99, Gou00]. We intend to further develop such techniques. Finally, another technique we intend to explore concerns contextual verification: verification via a "testing context" which specifies all the interesting stimuli (depending on the application). The result of the analysis will not be as conclusive as checking equivalence, but it may be a way to increase confidence or to discover errors.

Proving spatial properties In this theme, we focus on the application of model-checking and equational reasoning techniques to automate the proof of spatial-temporal properties of processes expressible in fragments of the logics resulting from the work in WP2.

Co-inductive techniques This theme will focus on the extension of co-inductive techniques to the case of mobile distributed systems. We intend to develop co-inductive techniques for Ambient-like process calculi. In particular we will exploit these results to provide "up-to" techniques for these formalisms. We also intend to investigate algebraic theories and axiomatizations of languages for mobile distributed processes. We shall begin with the finite part of Ambient calculi. Finally, the use of HD-automata and their generalization (Task 1.1), will allow the efficient exploration of the state space of systems which are a priori infinite.

Task 1.3: Prototype and case studies

The goal of this task is the design and implementation of an environment which supports specification and verification of properties of network applications. We will also consider some illustrative case studies to test the effectiveness of the verification environment.

Verification environment: tool development The proposers of this project have already developed some prototypical verification environments [15, 51]. However, both verification environments need restructuring and most probably a complete reimplementation. Hence, a main challenge of this task is the collaboration among the sites of the projects in designing and implementing a common verification environment based on the automata model of Task 1.1. We expect to apply our foundational results on the basic model and the proof techniques to drive the design and the implementation of the environment.

The functionalities of the verification environment will be checking of equivalences and preorders, minimization procedures, model checking, and symbolic analysis. We also consider modular extensions of the verification environment towards particular application areas; this may call for additional specialized modules (for instance to handle verification of security protocols). The novelty and added value of this approach is in the integration of these aspects into one environment, and the establishment of an efficient internal representation in terms of basic model developed in Task 1.1. Corresponding issues have been solved in the context of ordinary process algebra (e.g. the Meije toolkit based on the FC2 format) but to date remain elusive for distributed and mobile process calculi. We also need to address the issues related to the development of user interface. The aim is to have an interface easily used by all partners and organizations of similar competence.

Case Studies This theme will focus on the analysis of some case studies. In particular we will assess our verification environment by considering the problem of certifying security properties. The case studies selected will be about non-interference and access control schemes and symbolic analysis of security protocols.

Workpackage 2: Specifications

The goal of this workpackage is to develop a logical framework to support the specification of mobile concurrent systems, and able to address difficult features of highly mobile and dynamic applications. To that end, we will develop new logics and proof systems to support formal reasoning about of such systems, and related verification tools.

Task 2.1: Logics for systems with spatial and temporal structure

In this task, we want to systematically study logical formalisms able to handle dynamic and structural properties of models of mobile computation. We intend to study the interpretation of such logics with relation to several different models of mobile computation like process calculi and the syntax-free models to be developed in WP1.

Base logic We shall take as starting point some system of temporal logic with fixed-point operators, and study its extension with intensional connectives

aimed at observing the structural composition of states. We expect that at least part of such primitive structural connectives will arise as counterparts of the state constructors (or "controls") available in the underlying models, but the identification of the concrete primitives will also be a major goal of the research to be carried out.

Diverse notions of space and data domains Different applications may require different notions of space or structure. For example, a forest of trees is well suited to represent hierarchies of administration domains inside wide area networks. Labelled graphs provide an alternative basic structure for describing the topology (spatial distribution and connectivity) of networks. We shall investigate logics with respect to several general spatial structures like these and others. We shall also consider richer data domains, and not just names as usually done in π-calculi (e.g. abstract data-types or at least constructor-based algebraic data-types).

Proof systems We shall design proof systems in two related directions. On the one hand, we will develop sequent-based proof systems where entailment is interpreted in terms of the properties expressible by logical formulas and validity taken relative to a class of models. On the other hand, building upon the latter systems, we intend to study proof systems for certain specific models, targeted at proving properties of concrete process terms.

Verification framework We shall build frameworks for semiautomatic verification which are both usable in practice and general enough to cope adequately with the difficult system features which will be central to the highly mobile and dynamic applications envisaged in the future. In previous work we have found that a framework based on sequent calculus extended with natural deduction-like mechanisms to support inductive and co-inductive reasoning provides a good starting point. The challenges are both to extend the framework in a systematic way to handle rich features like exceptions, links, resources, names, dynamically created object and references. Equally methods, for instance from the field of automated induction, must be taken in to support the level of automation, and hence the amount of detail needed to be exposed to the user of an interactive verification tool. To address this issue we intend to investigate the extent to which rippling-like techniques can be used in collaboration with instance-checking and well-foundedness checking to automate induction schema generation and verification.

Task 2.2: Expressiveness

This task will focus on accessing the expressiveness of the base logics and identifying suitable high-level extensions. The ultimate goal is to develop high-level modalities, expressible in the base logic, which help in writing and proving properties by exploiting specialized proof techniques, and enhance the usability of the logical framework and associated tools.

Decidability issues The semantics induced by the logics of space is intensional, and therefore much finer than that induced by standard operational semantics. This is in striking contrast with the situation for "classical" modal logics, for which characterization results relating logics and operational equivalences (typically forms of bisimilarity) exist. We want to investigate the consequences of this intentionality on decidability problems for the logics. First we shall study whether the equivalence induced by the logics is decidable on the full Ambient calculus and on the π-calculus. Another issue to address is the impact of variations on the semantics of the logics on questions of decidability and intensionality. We aim at defining spatial logics that agree with operational semantics, as classical modal logics with bisimilarity. We also intend to isolate and study interesting fragments of both the base logic and concrete operational models, and establish completeness and decidability for them.

High-level extensions Due to the level of generality being sought, one expects that the primitive constructions of the base logics turn out to be of quite low level. For example, the modal μ-calculus directly applied to the π-calculus results in awkward formulas, even though the system is in principle very expressive. We shall determine relevant high-level primitives and associated proof principles. This will be done by studying concrete application problems and generalizing from them, in close connection with the task on case studies. Typically, such higher-level constructs will allow the direct expression of "user-level" properties like the statement of a system wide routing policy and the querying the access rights of a particular mobile agent. More basic examples of primitives could be: assertions of message uniqueness or about knowledge encoded in different parts of a distributed system, or about a few built-in data types.

We also want to investigate to what extent spatial logics can be used to describe dynamic coordination policies of an evolving (both in space and in time) network, when the underlying model is a forest (or graph) of trees. The ultimate goal is to develop high-level logical primitives to describe properties of the temporal and the spatial evolution of a variety of networks.

Task 2.3: Tools and case studies

Tools We shall build tools to support a logical framework for a version of the spatial logic, integrating theorem proving and algorithmic techniques (mainly type- and model-checking) developed in other tasks. Such tools will be implemented on top of an existing verification tool, but we also intend to explore general purpose theorem-proving technology for semiautomatic user-assisted verification. It will read-in system specifications, written in some high-level idiom of the logic, and then allow the user to interactively and/or mechanically validate properties. The tools should be configurable so as to be able to handle several different models, namely we intend to develop generic verification tools which will handle several operationally defined languages, including process calculi like π-calculus or the Ambient calculus, in a systematic way. Progressive experimentation with the tool during the course of the project will help the identification

of typical proof patterns, tactics, and of high-level extensions mainly studied in Task 2.2.

Case studies Closely connected with Task 2.2, we aim to develop specific theories and related proof principles, together with its implementation on the verification tools. Possible case studies involve distributed algorithms, including locking and security protocols and self-stabilizing algorithms, and smart-card related applications, for instance interference problems on executions of concurrent threads. Another area to look for applications is in distributed games. We also intend to pick some of the already mentioned process calculi, and a small object-oriented concurrent language, which may also be considered for demonstration purposes.

Workpackage 3: Types

Type systems provide a robust and attractive framework for constraining processes' behavior: they are well-understood in the realm of traditional sequential programming languages, support modular reasoning and provide a criterion to decide whether a process is well-behaved (in the sense that a well-typed process will always have an acceptable behavior). The purpose of this workpackage is to explore verification techniques for mobility based on types.

Task 3.1: interference and access control

In this task we will design novel type systems for mobile code. The emphasis will lie on type systems that allows us to control interferences among processes, and their access to resources. Below are the main themes that we intend developing.

Classification of interferences In formalisms for mobile distributed systems, interference among processes can appear in new and unexpected forms; see for instance the study of interferences in the Ambient calculus [54]. We wish to understand better the nature of these interferences, making the informal classification between *plain* and *grave* interferences in [54] into a formal definition that is applicable to several formalisms; refining this classification; studying the consequences of these interferences on the behavioural theory of processes.

Resource allocation We will develop, in the context of a small but realistic language for mobility, type systems that enforce security policies expressed in terms of resource allocation. Typically, the type system should prevent agents that penetrate a local network to consume more resources than allocated.

Declassification We intend to enrich capability-based types that exist for distributed versions of π-calculus and coordination calculi (Linda-like) with mechanisms supporting declassification, i.e., the ability to downgrade some confidential information temporarily or to dynamically modify protection domains.

This will permit to change the protection domain of the components of a network applications when the application is running. This feature turns out to be essential when dealing with applications for e-commerce, and will probably require a more extensive dynamic type checking.

Secure composition of components Another issue in which we believe types can be important is that of secure composition of components. This has to do with the combination of various devices, resources, and applications, ranging from stationary workstation and databases to Personal Digital Assistants (PDAs). Since some components can be dynamically downloaded from the network, the secure composition of components is a crucial activity. Several approaches have been proposed, but to the best of our knowledge, even the formal assessment of the security requirements for the composition of components is not well developed. In order to address this point we will first restrict ourselves to a simple language, a process calculus with mobility primitives, where components and their composition primitives are naturally expressed. We intend to use secure types (either in form of Interference types or capability-based types) to specify and enforce the security requirements of component composition.

Task 3.2: Integration of types with operational and logic techniques

Existing type systems for mobility have been studied in isolation and designed on basic foundational calculi, most frequently dialects of the π-calculus. The same will hold for the new type systems that will be developed under Task 3.1.

This task 3.2 is about integration. This will take several forms: integration of type systems; comparison of verification techniques and constructs, combination of techniques.

Operational and logical techniques for typed calculi We wish to extend the techniques developed in WP1 and WP2 to typed calculi. This is a hard problem because the classical notions of observation, in use in untyped calculi, are far too strong in typed calculi. Some results on this problem exist, but again, they only apply to a few specific calculi and type systems. Our aim is to arrive at a general, uniform, framework in which several type systems can be accommodated, and in which the existing results can extracted as special cases. We will also study the use of type information throughout automatic verification in order to increase the efficiency of the techniques in WP1 and WP2, like those for minimization and model checking.

Expressiveness We are interested in establishing results relating the expressive power of spatial logics with that of type systems. Since both types and formulas denote sets of processes (cf. Propositions-as-types), we may start by investigating embeddings of particular type systems into theories of the more general logics. Such embeddings may then be used to derive other (eg. more precise, informative or abstract) logical characterisations of the properties ensured by the type systems we started with. On the inverse direction, but along

the same lines, we intend to isolate logical fragments for which the inhabitation problem is decidable, and that therefore yield in a natural way type-checking systems and algorithms.

Space in types Building on the study of the interpretation of types as logic formulae, we intend to design new, more intensional, type operators, which would be based on the spatial logics operators introduced in WP2. This will permit to introduce more dynamical aspects, acting both over time and space, into the type system. The new type system will allow us to encode, by using types, temporal and spatial dependences among protection domains. As an example of such properties, a type could express the fact that a mobile component that executes over a host can perform remote outputs only until it does not perform local inputs.

Integration of type systems An important objective is the integration of different type systems and the related proof techniques. This will allow us to obtain type systems in which different security aspects (such as protection of certain resources, access to sites) can be specified. This will also lead us to develop methods for modular proofs on large programs, in which proofs follow the structure given by the type abstractions. This work may be initially carried out in π-calculus-like formalisms; it will then be extended to more ambitious languages, for instance some small concurrent object-oriented language.

Approximation techniques Finally, a fall-out of the research on decidability problems for π-calculus-like formalisms (Tasks 1.2 and 2.2) should concern the design of various conservative techniques based on types to analyse π-calculus processes. Currently a number of 'type systems' have been proposed to establish properties such as absence of deadlock in π-calculus processes [IgKo01]. These systems currently perform a very rough analysis of the dynamics of the system. Typically, to type the replication of a process P try to type P. We expect that a fine study of the decidability issues for the π-calculus will shed light and justify various approximation techniques in this area. We also project to apply our techniques to the analysis of sizeable pieces of code written in experimental languages such as JoCaml or Pict.

Task 3.3: practicality and scalability

Type-inference algorithms We will study the practicality of our type-based techniques by devising type-inference algorithms that verify whether a given process is type-correct. While input from the user will be required (if only to indicate that he wishes to constrain some process's capabilities), as much information as possible should be synthesized by the algorithm.

Advanced programming constructs We will then study the scalability and robustness of our approach by incrementing the foundational calculi on which

the type systems have been designed with more advanced programming constructs, so to move towards realistic programming languages. We are especially interested in the extensions to concurrent object-oriented languages.

Task 3.4: Case studies and tool development

Case studies The applicability of our type systems will be assessed by a set of case studies dealing with examples from the fields of e-commerce Typically, such case studies involve mobile communicating applets with limited resources and time constraints. They are thus particularly well-suited to serve as a test-bed for the type systems for control of interferences and consumption of resources that we intend to develop. Other case studies will be borrowed from the field of Supported Collaborative Works. We aim carrying out the analysis of some of the case studies with our tools.

Implementations We will study implementations for the type inference and type checking algorithms. The latter will be done in connection with the tools developed under WP1 and WP2. In general, effort will be devoted to exploiting results from WP3 in order to make the tools in WP1 and WP2 more usable, for example by including type information in the analysis algorithms. Another effort, in connection with Task 3.2, will focus on extending the tools to accommodate typed processes.

Workpackage 4: Management, Assessment and Dissemination

Task 4.1: Management

In this task we manage the project as further described in Section 9.7. It involves the project coordinator, workpackage and site leaders and steering committee. There are no formal deliverables from this task, but activities will be reported in yearly progress reports.

Task 4.2: Assessment

Self-assessment and evaluation of progress in the project will be conducted on a yearly basis. The criteria will be related to the project goal of making it possible to automatically or with considerable automatic support formally analyse the distributed mobile systems that we target. Since our goals are to bring out principles and scientific foundations for such analyses rather than implementing a fully fledged industrial tool, measure of progress must be on the potential impact of our work. We therefore separate the criteria for self-assessment in two classes: results and dissemination.

Results are the scientific and technical knowledge gained within Workpackages 1-3. The results are reported in the deliverables and shall be evaluated in relation to the detailed plans for these workpackages. Particular attention

shall be put on what kind of impact the results may have for the analysis of mobile distributed systems. Thus, the criteria shall be:

- the definition of automata-based models which give meaning to a number of features of, and operations on, mobile processes
- the development of verification algorithms and proof techniques for these models
- the definition of logics capable of handling dynamic and structural properties of mobile distributed systems
- the development of proof systems to support formal reasoning about of the logics of interest
- the definition of new type systems that meet the need of global computing concerning security, control of interference, control of resources
- the definition of appropriate behavioural and operational theory for the languages equipped with such type systems
- the evaluation of the models, logics, and types, in case studies as well as in tools for mechanical manipulation

Dissemination is how well the project has succeeded in spreading the results to the rest of the community. This can be evaluated by considering the success of publishing or other dissemination efforts (see Task 4.3) and by considering the way in which results from the project are received and have an impact.

Self-assessment procedure A firm self-assessment is needed at the Workpackage level, since results and dissemination to some extent must be related to rather specialised scientific and technical communities. Each Workpackage leader is therefore responsible for a yearly written internal assessment of the corresponding Workpackage, according to the criteria above. The WP leader may if he deems it relevant appoint an external reviewer to assist in this task; however, it is the WP leader who writes the report. The report should be available well in time for the yearly Steering Committee meetings.

The WP self assessments are discussed at the yearly Steering Committee meetings and this discussion forms the basis of the yearly project self-assessment, a yearly formal deliverable written by the coordinator. This self-assessment also forms the basis for the decision by the Steering committee to steer the project activities.

In the Self-Assessment plan, a deliverable due at month 6, the procedure and criteria will be elaborated.

Task 4.3: Dissemination

Publications All participants will strive to disseminate their results by publishing in the best journals such as IC, MSCS, TCS, JACM, FAC, and confer-

ences such as ICALP, CONCUR, LICS, ETAPS, CAV, POPL, ACM conf. on communication and computer security, on the topic of verification and mobility.

We will publish parts of the Final Project Report in an international journal, and/or present them at an international conference.

A web site will be maintained containing: all publicly available results, available positions, minutes of discussions at workshops.

Workshops As main project events, we propose 4 *Workshops*, one every 12 months. At each workshop (but the first) a few external speakers will be invited, to allow us comparisons with what is being done outside the project, and for useful feedback. We plan to have 1 external speaker from academy and 3 or 4 external speakers from industry. Further, in each workshop with external observers at least half a day will be devoted to tutorial presentations of techniques and tools developed in the project, specifically aimed at the external observers.

The workshops will be the main occasion where the progress and the joint research among the partners will be presented and discussed. They will also be the occasion for discussing the overall situation of the project. The first workshop will occur at the very beginning of the project life and will be largely formed by tutorial talks, so that the project participants get to know each other better. Summaries of the scientific material presented at the workshops will be collected in informal publications, distributed to the participants and kept as project documents.

School A school will be organised together with the last workshop. The school will include basic courses on process calculi for mobile processes, proof techniques, logics, type systems for them, tools presented by members of the project or, where useful, by external specialists. The school has the main goal of attracting young students and researchers from outside the area of process mobility, and therefore it will be open to post-graduate students and post-doctoral researchers of both project partners and external institutes.

The deliverables in this task will be yearly summaries of the activities, including summaries from the workshops and school, the dissemination and use plan at month 6, and a final technology implementation plan.

9.2 Workpackage list

WP No	WP title	Lead contractor No	Person-months	Start mo.	End mo.	Deliverable No
1	Models	4	165 (117)	1	36	1-3
2	Specifications	2	120 (92)	1	36	4-6
3	Types	3	81 (64)	1	36	7-9
4	Management	1	15 (3)	1	36	10-18

Note: the personmonths represent the total effort from all partners, including effort not directly paid by the commission. The paid effort, which is further

detailed in the CPF, is given in parenthesis. A breakdown of paid and unpaid personmonths per workpackage and year is given in Table 1.

9.3 Workpackage descriptions

Whole project:

	Y1	Y2	Y3	Sum
UU:	18 (12)	29 (18)	35 (24)	82 (54)
FFCT:	45 (33)	45 (33)	45 (33)	135 (99)
INRIA:	23 (23)	23 (23)	23 (23)	69 (69)
PISA:	30 (12)	35 (18)	30 (24)	95 (54)
Total:	116 (80)	132 (92)	133 (104)	381 (276)

WP1 (Models):

	Y1	Y2	Y3	Sum
UU:	16 (12)	15 (10)	15 (10)	46 (32)
FFCT:	14 (11)	14 (11)	14 (11)	42 (33)
INRIA:	6 (6)	6 (6)	6 (6)	18 (18)
PISA:	18 (7)	23 (12)	18 (15)	59 (34)
Total:	54 (36)	58 (39)	53 (42)	165 (117)

WP2 (Specifications):

	Y1	Y2	Y3	Sum
UU:	0 (0)	6 (4)	9 (7)	15 (11)
FFCT:	24 (18)	24 (18)	24 (18)	72 (54)
INRIA:	6 (6)	6 (6)	6 (6)	18 (18)
PISA:	5 (2)	5 (3)	5 (4)	15 (9)
Total:	35 (26)	41 (31)	44 (35)	120 (92)

WP3 (Types):

	Y1	Y2	Y3	Sum
UU:	0 (0)	6 (4)	9 (7)	15 (11)
FFCT:	6 (4)	6 (4)	6 (4)	18 (12)
INRIA:	10 (10)	10 (10)	10 (10)	30 (30)
PISA:	6 (3)	6 (3)	6 (5)	18 (11)
Total:	22 (17)	28 (21)	31 (26)	81 (64)

WP4 (Management):

	Y1	Y2	Y3	Sum
UU:	2 (0)	2 (0)	2 (0)	6 (0)
FFCT:	1 (0)	1 (0)	1 (0)	3 (0)
INRIA:	1 (1)	1 (1)	1 (1)	3 (3)
PISA:	1 (0)	1 (0)	1 (0)	3 (0)
Total:	5 (1)	5 (1)	5 (1)	15 (3)

Table 1: Breakdown of person months per year and workpackage.

Workpackage Description

Workpackage number:	1	Start date or starting event:	1

Participant number:	1	2	3	4
Person-months per participant:	46 (32)	42 (33)	18 (18)	59 (34)

Description of work

The workpackage comprises three tasks:

Task 1.1: Automata with operations and substitutions

We shall develop an automata-like model based on general name substitutions. The model will include data constructors like encryption and various forms of spatial composition. We will proceed by isolating and studying interesting fragments of the model and establishing completeness and expressivity results for them.

Task 1.2: Proof techniques

We will study which properties are of practical interests and which proof techniques can be consequently designed. The proof techniques will be both automatic (model checking, bisimilarity checking) and semiautomatic (coinduction proofs, symbolic execution).

Task 1.3: Prototype and case studies

The foundational results will drive the design and development of an effective semantic based-verification environment which will support specification, design and and verification of network applications.

Deliverables

1 – Automata with name fusions and operations; verification via semantics equivalence and proving spatial properties; tool implementation: automata with permutations.

2 – Automata with substitutions and models for spatial logics; symbolic execution; tool implementation: extended automata.

3 – Verification with extended automata and co-algebraic techniques; verification and case studies.

Milestones and expected results

Year 1 (+12) – Production of deliverable 1.
Year 2 (+24) – Production of deliverable 2.
Year 3 (+36) – Production of deliverable 3.

Interactions with other WPs

The logics of Task 2.1 will be interpreted in the syntax-free models of Task 1.1. Tools developed in Task 1.3 will be used as a basis for the case studies and tools of Task 3.4.

Workpackage Description

Workpackage number:	2	Start date or starting event:	1

Participant number:	1	2	3	4
Person-months per participant:	15 (11)	72 (54)	18 (18)	15 (9)

Objectives

To develop a logical framework and a supporting tool for the specification and verification of correctness properties addressing both the structure and the behaviour of concurrent mobile systems.

Description of work

The workpackage comprises three tasks:

Task 2.1: Logics for systems with spatial and temporal structure

This task will develop new logics to support the verification of both structural (spatial) and behavioural (temporal) properties of distributed mobile systems. The logics will be interpreted in several different models of mobile computation like process calculi and the syntax-free models to be developed in WP1.

Task 2.2: Expressiveness

This task will study the expressiveness of the logics. In order to enhance the usability of the logical framework and associated tools, higher-level extensions and corresponding specialized proof principles will be developed.

Task 2.3: Tools and case studies

This task will build tools to support a logical framework for a version of the proposed logic, integrating theorem proving and algorithmic techniques (mainly type- and model-checking) developed in other tasks. Those tools will be applied to several case studies.

Deliverables

4 – Base logics and proof systems.
5 – Verification framework: decidability results, algorithms and extensions.
6 – High-level extensions, case studies and implementations

Milestones and expected result

Year 1 (+12) – Production of deliverable 4.
Year 2 (+24) – Production of deliverables 5.
Year 3 (+36) – Production of deliverables 6.

Interactions with other WPs

The logics of Task 2.1 will be interpreted in the syntax-free models of Task 1.1. Verification framework developed in Tasks 2.2 and 2.3 will integrate model- and type-checking techniques developed in Task 3.3 and Task 1.2.

Workpackage Description

Workpackage number:	3	Start date or starting event:	1

Participant number:	1	2	3	4
Person-months per participant:	15 (11)	18 (12)	30 (30)	18 (11)

Description of work

Task 3.1: Interference and access control

In this task we will design novel type systems. The emphasis will be on types that control interferences among processes and their accesses to the system resources. Most notably, we will develop types that: enforce security policies for resource allocations; support declassification; allow secure composition of components.

Task 3.2: Integration

This task will

— integrate different type systems and the related proof techniques;

— integrate types into the techniques based on automata and logics from WP1 and WP2, so to be able to apply such techniques to typed process, but also in order to increase efficiency of the techniques by taking into account some type information.

— design new, intensional, type systems based on the spatial logics of WP2.

A fall-out of this research will be results of comparison and of expressiveness of different techniques and constructs.

Task 3.3: Practicality and scalability

This task will study how to scale up the type techniques to practical programming languages, and will device algorithms for the automatic verification.

Task 3.4: Case studies and tool development

This task will develop implementations for the type techniques, and will apply them to case studies, mainly from e-commerce.

Deliverables

7 – Novel type systems
8 – Integration and algorithms
9 – Advanced programming constructs and implementations

Milestones and expected result

Year 1 (+12) – Production of deliverable 7.
Year 2 (+24) – Production of deliverable 8.
Year 3 (+36) – Production of deliverable 9.

Interactions with other WPs

The work in WP1 and 2 during the first half of the project will guide part of Task 3.2

Task 3.4 uses the tools and implementation developed in WP1 and 2 as a basis.

Workpackage Description

Workpackage number:	4	Start date or starting event:	1

Participant number:	1	2	3	4
Person-months per participant:	6 (0)	3 (0)	3 (3)	3 (0)

Objectives

The main activities in this workpackage are: coordination and evaluation of the technical activities in the project; preparation and distribution of reports; dissemination of knowledge acquired; liaison with external persons (other research groups, EU community, etc.); organisation of project meetings and the project School.

Description of work

Task 4.1: Management The Project Coordinator is responsible for: administrative and management matters, including the interface with EU, and the writing of the progress reports.
A Steering Committee (SC), chaired by Project Coordinator, is the main decision body for the project and will meet at least once per year. SC monitors the whole activity of the project, including: progress and weaknesses of each task; project management; organisation of workshops.
Each workpackage has a leader who has the responsibility of reporting on that workpackage, identifying progresses and weaknesses; coordinating the activities and assessment within the workpackage; coordinating the activities of the workpackage with those of the other workpackages.
Task 4.2: Assessment Yearly assessments of each workpackage according to defined criteria are conducted by work package leaders and are discussed by the steering committee. Based on this a project wide assessment is conducted by the project coordinator. Success criteria relate to scientific and technical progress towards the goals, and potential impact.
Task 4.3: Dissemination An annual workshop will be organised, with participation of some external speaker from academy and industry. A School will be organised together with the last workshop. A web site will be maintained containing: all publicly available results, available positions, minutes of discussions at workshops. All participants will strive to disseminate their results by publishing in the best journals and conferences

Deliverables

10 – Dissemination and use plan
11 – Summary of Dissemination activities year 1
12 – Summary of Dissemination activities year 2
13 – Summary of Dissemination activities year 3
14 – Technology Implementation Plan
15 – Self-assessment Year 1
16 – Self-assessment Year 2
17 – Self-assessment Year 3
18 – Self-assessment plan

Milestones and expected result
Year 1 (+1) – First project workshop.
Year 1 (+6) – Production of Deliverables 10 and 18.
Year 1 (+12) – Second workshop. Production of Deliverables 11 and 15
Year 2 (+24) – Third workshop. Production of Deliverables 12 and 16
Year 3 (+36) – Fourth workshop, School. Production of Deliverables 13, 14 and 17.

9.4 Deliverables list

See Table 9.4.

No	Deliverable name	WP no	Lead part.	Estimated manmonths	Type	Security	Month
1	Automata with name fusions and operations; verification via semantics equivalence; and proving spatial properties; tool implementation: automata with permutations.	1	Pisa	48	report	Pub	12
2	Automata with substitutions and models for spatial logics; symbolic execution; tool implementation: extended automata.	1	Pisa	50	report/p	Pub	24
3	Verification with extended automata and co-algebraic techniques; verification and case studies	1	Pisa	50	report/d	Pub	36
4	Base logics and proof systems	2	FFCT	30	report	Pub	12
5	Verification framework: decidability results, algorithms and extensions	2	FFCT	30	report/p	Pub	24
6	High-level extensions, case studies and implementations	2	FFCT	34	report/d	Pub	36
7	Novel type systems	3	INRIA	36	report	Pub	12
8	Integration and algorithms	3	INRIA	23	report	Pub	24
9	Advanced programming constructs and implementations	3	INRIA	23	report/p	Pub	36
10	Dissemination and use plan	4	UU	2	report	Pub	6
11	Summary of Dissemination Year 1	4	UU	1	report	Pub	12
12	Summary of Dissemination Year 2	4	UU	1	report	Pub	24
13	Summary of Dissemination Year 3	4	UU	1	report	Pub	36
14	Technology Implementation Plan	4	UU	2	report	Pub	36
15	Self-assessment Year 1	4	UU	1	report	Pub	12
16	Self-assessment Year 1	4	UU	1	report	Pub	24
17	Self-assessment Year 1	4	UU	1	report	Pub	36
18	Self-assessment plan	4	UU	1	report	Pub	6

Table 2: Deliverable list. "report/p" means report and prototype; "report/d" means report and demonstrator.

9.5 Project planning and timetable

Year 1	Year 2	Year 3
1.1		
1.2, 1.3		
2.1		
	2.2, 2.3	
3.1, 3.2		
	3.3, 3.4	
4.1, 4.2, 4.3		

Tasks	Years	Sites	Dependencies from tasks	Deliverables
1.1	1,2	1,2,4	2.1	1,2
1.2	1,2,3	1,2,3,4	1.1, 2.1, 2.2	1,2,3
1.3	1,2,3	1,4	1.1, 1.2, 2.3, 3.4	1,2,3
2.1	1,2	2,3,4	-	4,5
2.2	2,3	1,2,3,4	1.1, 1.2, 1.3, 2.1, 2.3	5,6
2.3	2,3	1,2,4	1.2, 2.1, 2.2, 3.2	5,6
3.1	1,2	3,4	-	7
3.2	1,2	3,4	3.1, WP1, WP2	7,8
3.3	2,3	2,3,4	3.1, 3.2	8,9
3.4	2,3	1,3,4	3.1, 3.2, 3.3, WP1, WP2	9
4.1	1,2,3	1,2,3,4	-	-
4.2	1,2,3	1,2,3,4	-	15, 16, 17, 18
4.3	1,2,3	1,2,3,4	-	10,11,12,13,14

9.6 Graphical presentation of project components

See Figures.

Gannt chart for section 9.5

	Month	1	6	12	24	36
WP1	1.1			D1	D2	
	1.2			D1	D2	D3
	1.3			D1	D2	D3
WP2	2.1			D4	D5	
	2.2				D5	D6
	2.3				D5	D6
WP3	3.1			D7	D8	
	3.2			D7	D8	
	3.3				D8	D9
	3.4				D8	D9
WP4	4.1					
	4.2					
	4.3					
	4.4		D10	D11	D12	D13,14
		W1		W2	W3	W4,S

Double vertical bars signify the start and end of each task. Dn is Deliverable n, Wn is Workshop n, S is School.

9.7 Project management

Steering Committee. This is the main decision body for the project. The Steering Committee will consist of the leaders of the participant teams in the project, namely: Luis Monteiro (FFCT), Davide Sangiorgi (INRIA), Björn Victor (UU), Ugo Montanari (PISA), plus the Project Coordinator (Joachim Parrow, UU), who will act as chairman. The Steering Committee is responsible for monitoring the whole activity of the project. The Steering Committee will meet at least once a year, after each Workshop. These meetings will be the occasion for discussing: (1) progress and weaknesses of each task; (2) mobility of people within the consortium; (3) project management; (4) organisation of workshops and, possibly, of other events; (5) other issues. Minutes of each of these discussions will be written and circulated in the form of an electronic newsletter.

Progress reports. At the end of each year a progress report will be produced, which includes: i) technical achievements during the reporting period; ii) major issues of problems which might delay activities, and corrective actions undertaken; iii) serious problems that cannot be solved internally to the project, but that require attention from the EU services; iv) meetings attended by project partners; v) plans for the activities of the following year.

Workpackage Leaders. The workplan of the project is divided into 3 proper workpackages, as explained in the project workplan. Each workpackage has a *leader*. These are: Models: Ugo Montanari (PISA); Specifications: Luis Monteiro (FFCT); Types: Davide Sangiorgi (INRIA). A Workpackage Leader has the responsibility of: i) reporting on his workpackage (this includes writing the workpackage contribution to the progress reports and dissemination summary); ii) identifying the progresses made and the weaknesses; iii) coordinating the tasks within the workpackage; iv) coordinating the activities of the workpackage with those of the other workpackages. v) timely production of workpackage deliverables.

In case weaknesses are found, adjustments will be discussed at the Steering Committee meetings.

A fourth workpackage is devoted to management, assessment and dissemination; here the project coordinator is the leader.

Task leaders will be appointed for each of the tasks in which workpackages are divided. A task leader, appointed by the leader of the corresponding workpackage, is responsible for the timely progress of the corresponding task, and reports to the coordinator of the workpackage to which the task belongs at least twice per year. Events that could effect the progress of the task will be reported immediately.

Project Coordinator. The Project Coordinator is Joachim Parrow (UU). He is responsible for the administrative and management matters, acting as principal contact person with the Commission. He will write the progress reports, using the contributions received from the Workpackage Leaders. Both the Project Coordinator and UU have ample experience in research management.

Yearly review. At the yearly reviews, the progress of the project and the

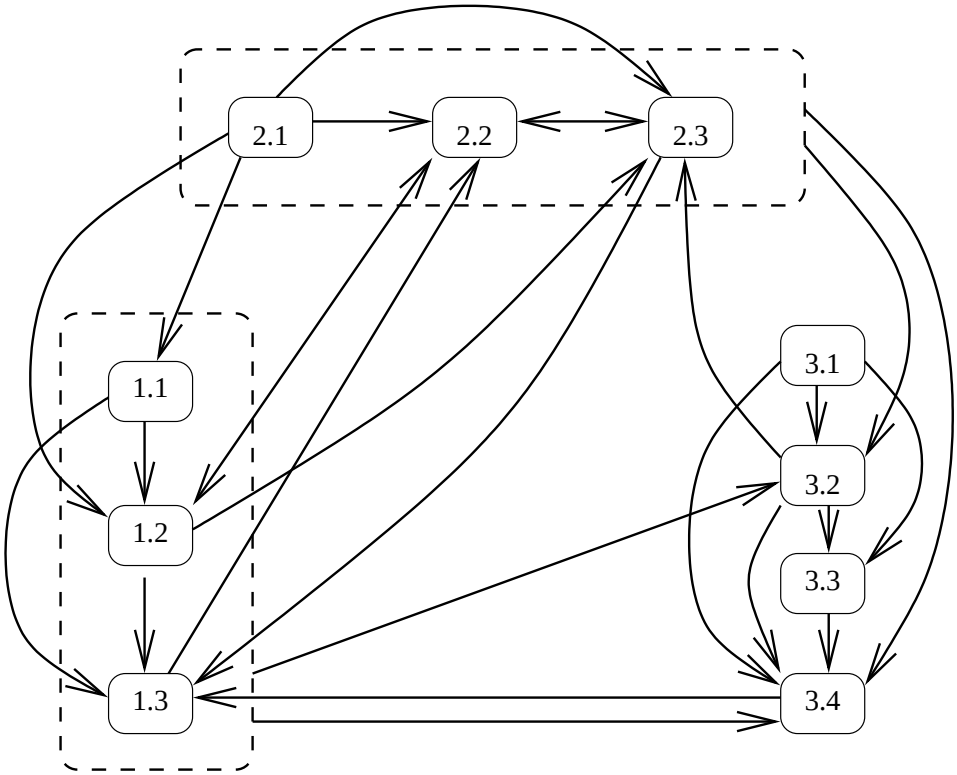

Figure 1: Dependencies between tasks

dissemination of its results will be critically reviewed. Depending on the progress and the results achieved, some change in the programme may be proposed.
Site leaders. Each site leader is responsible for the scientific and organizational management of his team. He must ensure that the contributions of his team to the progress reports are delivered in time to the corresponding workpackage leaders. He must organize a Workshop. He is responsible for advising the project coordinator on progress or weaknesses as they emerge.
Communications among participants. Good communications among the participating teams will be ensured by: i)the participation at the Workshops and School; ii) visits among partners; iii) exchange of young researchers. Www pages will be set up that contain information regarding project activity, publications, available positions, managing information, and minutes of discussions at workshops. Electronic mailing lists will the primary medium for both technical and administrative exchanges: one for each workpackage, but also possibly other ad-hoc lists for important problems or topics that will emerge.
Visits We foresee several long-term (some weeks to a few months) exchanges of researchers among participants especially in the initial phase of the project, to foster cooperation among sites. Due to the cost of these exchanges and of frequent traveling of senior participants to carry on the advanced research necessary to the success of the project, sufficient resources are needed for travel and subsistence, and have been exposed in form A4.
Conflict resolution
Local conflicts A conflict that is wholly internal to one of the sites shall be resolved by the respective site leader.
Managerial conflicts A conflict where the issue is managerial or relates to contacts between the project and the commission, and has no scientific content, shall be resolved by the project coordinator. If the coordinator or any party in the conflict deems it necessary, the coordinator shall first confer with the project officer in Brussels.
Global conflicts A conflict whose nature is scientific and has impact on the direction of the work in the project that is not local to one site shall be resolved as follows. The project coordinator is responsible for carrying out the procedure described below. First, the coordinator collects information on the conflicting views and presents them impartially to the steering committee. The coordinator must make sure that all members of the committee are informed and have understood the situation. The coordinator then initiates a discussion in the SC with the goal of obtaining a consensus. If a consensus cannot be obtained the matter is put to a vote, where each site carries one vote. The vote should only consider two alternatives, and the alternatives are formulated by the coordinator on the basis of the discussion. A decision is taken through a simple majority of the votes. In case of a tie the coordinator decides.

If the nature of the conflict is such that it can wait until the next yearly workshop then it is taken up at the steering committee meeting at that workshop. If it cannot wait then the procedure above may be carried out through email. If the coordinator or any two steering committee members wish, an extra steering committee meeting may be called.

If the category (local/managerial/global) of a conflict is ambiguous or questioned the it shall be resolved by the procedure for a global conflict.

10 Clustering

PROFUNDIS is particular interested in clustering (for scientific information exchange and collaboration and possibly also dissemination activities) with the following projects, where types play a key role: DARTS, MIKADO, MRG, MYTHS.

Clustering might also be possible with with PEPITO and, although less clearly at present, with AGILE and DEGAS.

11 Other contractual conditions

Travel outside EU Some of the major conferences in our area are sometimes located outside EU, notably in the USA and Japan. We shall attend these conferences as warranted by the need to disseminate and discuss our results and to appreciate recent advances in the global scientific community.

Invited Speakers For our workshops and for the concluding school we aim at inviting speakers of high international standing, in order order to learn from their expertise but also in order to inform them of our progress. The cost for this is reported under "other cost".

External collaborators The following 3 persons will effectively work on the project, as external collaborator of the INRIA site; the project will pay some money for their travels and subsistance: Davide Sangiorgi (at present at INRIA, but will move to Univerisity of Bologna in September 2002), Roberto Amadio (who is at Université de Provence), Daniel Hirschkoff (who is at Ecole Normale Superieure de Lyon.) Note however that Davide Sangiorgi will be employed by INRIA for part of the year 2002. Similarly, Mads Dam at the Royal Institute of Technology, Stockholm, will be an external collaborator to UU.

Audit certificates Audit certificates will be supplied as necessary; their cost is listed under "other costs".

Appendix A:Consortium description

The consortium is extremely well qualified to meet the objectives of the project. In fact all participants have a long and well-documented expertise in models and reasoning techniques for mobility. Also the partners complement each other very well: automated tools (UU), constraint logics, modal logics for space and time (FFCT), coalgebraic and automata-based techniques such as HD-automata (PISA), and types (INRIA).

Contacts among sites have already happened: UU and INRIA on algebraic theory of mobility [13] and on modal logics for mobility [21]; PISA and INRIA on coinductive proof techniques and HD automata [60, 44, 43] and on types for controlling interferences [54]; INRIA has studied Ambient logics [35], very similar to those introduced by the FFCT group [18, 17]. UU and INRIA participated to the EU projects CONCUR I and II; all partners participated to either CONFER or COORDINA Esprit working groups.

UU will contribute mainly to algorithms, prototype tools, and applications. UU will implement in WP1 some fully automated analysis algorithms: here UU through close collaboration with the Royal Institute of Technology and SICS has considerable experience. UU also has experience in the pi-calculus and associated modal logics, and (with SICS) has worked on applications related to security and authentication. Significant contributions will be relevant for WP2 and WP3 in tasks relating to verification methods and case studies.

FFCT will contribute mainly to a logical framework for the specification and verification of structural and behavioural properties of mobile distributed systems. This is the main topic of WP2, which FFCT coordinates, where also a tool for automatic and semiautomatic verification of correctness properties will be built. FFCT will collaborate in WP1 on models for the logic and on model-checking techniques to be integrated with the tool. In WP3, results relating the expressiveness of the logic with that of type systems for aspects like interference and access rights will be established.

INRIA's main contributions will be on WP 3 (which INRIA coordinates): types for access rights, interferences, information flow. INRIA has also competences in algorithms for automated analysis of cryptographic protocols and modal logics of space and time, formalisation of calculi of mobile processes into theorem provers. INRIA intends to develop and strengthen these competences, and will contribute also to WP 1 and WP 2. Some members of the INRIA team are presently working on verification of security for JavaCard in collaboration with industries. This activity will provide useful case studies to the project.

PISA coordinates WP1. HD automata [59] have been already used for verifying π-calculus case studies [51] within the JACK tool developed by Stefania Gnesi (who will collaborate with the project). PISA will contribute also to WP2 and WP3 with its experience about network-aware models [46], typed mobile processes [42] and symbolic analysis of cryptographic protocols [39]. Marco Pistore of IRST will collaborate with PISA and IRST will participate to the project workshops.

Uppsala University - UU

Uppsala University comprises three Disciplinary Domains: Arts and Social Sciences, Medicine and Pharmacy, and Science and Technology. Education and research is conducted in eight Faculties. There are more than 40 programs of study, over 1,200 independent courses, 37,000 students, and 2,500 graduate

students. Over 300 doctorates yearly. The University has some 5,500 employees, including about 3,000 teachers/researchers. Turnover: bSEK 3.3 in 1999. Research and graduate education turn over about bSEK 2. It is the oldest university in the Nordic countries;founded in 1477.

The Faculty of Science and Technology at Uppsala University, which also includes the Uppsala Institute of Technology, is the largest of the University's faculties in terms of turnover. The Uppsala Institute of Technology, UTH, with its some 2,000 students, is a major educational force in engineering. Continued needs in society for IT competence, lifelong learning, and internationalization are some of the tendencies addressed in the strategic plan the Faculty has made its own. The Faculty hosts large-scale strategic research programs and continues to be successful in attracting new external funding.

The department of Information Technology is one of the largest in Sweden and hosts several research groups. Profundis will be conducted mainly at the subdepartment of computer systems, where research is focussed around algorithms and their properties, programming languages, how to create efficient, fast and correct machine code, machine learning, distributed systems, automatic electronic negotiations and resource allocation. We host two national centers (ASTEC and ARTES) aimed at facilitating collaboration between academia and industry.

We shall collaborate with groups at Kungliga Tekniska Högskolan, KTH, and the Swedish Institute of Computer Science, SICS. Common for the groups at UU, KTH and SICS are an emphasis on algebraic formulations. For the parallel, mostly deterministic applications, this means functional programming with lambda calculi and similar formalisms as a theoretical basis. For distributed systems this means process algebras like CCS and the π-calculus. The research is both theoretical, ranging from fundamental issues of semantics and models to theory supporting methods for efficient implementation, and experimental, with compilers, and systems for automated verification. The groups have participated in several EU-projects, notably CONCUR I, CONCUR II, REACT, LOMAPS, CONFER, VerifiCard, and the network EXPRESS and working group CONFER II, and collaborated with the INRIA/Sophia and Pisa research groups several times.

Parosh Abdulla, professor in Computer Systems at Uppsala University, obtained a BSc in electrical engineering from the university of Sulaimaniyah, Iraq, in 1982. He obtained a PhD in Computer Science from Uppsala University in 1990. In 1997 he received the degree of *"Docent"* in computer science. Since 2000, he is working as a professor at the department of computer systems in Uppsala. He has also been working on a part-time basis at *Prover Technology AB* in Stockholm. His main research interests include algorithmic verification of infinite-state systems, parametrized systems, and real-time systems.

Bengt Jonsson, Professor in Computer Systems, Uppsala University. Ph.D. 1987 at Uppsala University, researcher at SICS 1988-1992, professor in Computer Systems, Uppsala University, since 1992. He is member of the science and technology branch of the Swedish Research Council and the director of the

national center for advanced software Technology (ASTEC). His main research interests are in the areas of formal methods , especially in connection with real-time and distributed systems, semantics of concurrent systems, and verification of concurrent systems.

Joachim Parrow(project co-ordinator) Professor in Computer Systems 1992, UU. Ph.D. 1986 at Uppsala University, Researcher and group leader at SICS 1986-1994 with several visits to Edinburgh 1986-1989, Professor at KTH 1994-2001, head of department 1997-2000. Joachim Parrow pioneered the development of the Concurrency Workbench (with Rance Cleaveland and Bernard Steffen) and of the π-calculus (with Robin Milner and David Walker), and his recent research activities are on developments and extensions of the π-calculus, with strong interests also on tools and applications.

Björn Victor (site leader) Senior lecturer at Uppsala University since 1999. Ph.D. 1998 at Uppsala University, researcher at SICS 1992-1995. Björn Victor is the principal author of the Mobility Workbench (MWB), the first automated tool for verification of pi-calculus. Research interests include expressiveness, verification and analysis of calculi for mobility and security.

Mads Dam (external collaborator) Reader (docent) in Computer Systems, KTH. Ph.D. 1990 at University if Edinburgh, researcher, since 1994 leader of the Formal Design Techniques Lab at SICS. Lecturer, KTH, since 1990, and since 1991 leader of the Parallel and Distributed Systems Lab at KTH. Dam's research focuses on program verification and, more recently, computer security, including access management, privacy, and proof-carrying code. Dam's main contributions are in the areas of compositional verification techniques, temporal logics, and code validation. Dam currently holds grants in the areas of program code verification for the Erlang programming language and the JavaCard platform, secure mobile code, and secure authorization. Selected references: [3, 4, 5, 6, 7, 8, 16].

Universidade Nova de Lisboa - FCT/UNL (FFCT)

FFCT (Fundaçao da Faculdade de Ciências e Tecnologia) is a foundation of the Faculdade de Ciências e Tecnologia of the Universidade Nova de Lisboa (FCT/UNL). The New University of Lisbon (UNL), created in 1973, is the youngest of the three state universities in Lisbon. It has over 10,000 students, distributed by several Faculties and Institutes, which are located in several places in and around Lisbon. The Faculty of Sciences and Technology (FCT) is a few miles away from Lisbon, on the South side of the Tagus river. It has over 5,000 undergraduate students, about 350 postgraduate students and a staff of some 700 people. The Computer Science Department was the first of its kind in Portugal and is one of its leading centres of research and teaching in Computer Science and Engineering (CSE). It employs around 60 people, hosts an undergraduate curriculum in CSE (the first in Portugal, since 1975) and offers two Master's courses, one in CSE and the other in Artificial Intelligence. The research is organized in two centres, the Centre for Informatics and

Information Technologies (CITI) and the Artificial Intelligence Centre. The FFCT participates in the project through the group on Programming Languages and Models (PLM) from CITI. The other research streams of CITI are Software Engineering, Parallel and Distributed Processing Systems, Large Scale Distributed Computing Systems, Computer Graphics, Media Processing, Visualization and Interaction, and Geographic Information Systems. The group on PLM has worked in the last decade on logic programming, coordination languages, metric-like semantics for concurrency, and logics for the specification and verification of mobile systems. The research was conducted in several national and EU projects, namely ALPES, INTEGRATION, COMPULOG and COORDINA, and in the working groups COORDINATION and COTIC.

Luis Monteiro Professor of Computer Science, FFCT. Ph.D. 1983 at New University of Lisbon. Member of the Centre for Informatics and Information Technologies, New University of Lisbon. Prior research: logic programming, concurrency, metric semantics. Current research interests: theory and applications of coalgebras, theory of mobile systems.

Luis Caires, Assistant Professor of Computer Science, FFCT. Ph.D. 1999 at New University of Lisbon. Member of the Centre for Informatics and Information Technologies, New University of Lisbon. Main research interests: theory of programming languages, specification and verification of concurrent and mobile systems.

Antonio Ravara, PhD in Mathematics by the Technical University of Lisbon, 2000. MSc in Applied Mathematics by the Technical University of Lisbon, 1996. BSc in Applied Mathematics by the University of Lisbon, 1991. Assistant Professor in the Section of Computer Science, Mathematics Department, Instituto Superior Tecnico, Technical University of Lisbon. Research on theory of concurrent computation, in particular on mobile calculi.

Selected publications: [17, 18, 19, 20].

INRIA Sophia Antipolis - INRIA (INRIA)

INRIA (Institut National de Recherche en Informatique et en Automatique) is dedicated to basic and applied research in information technology. It is a leading research institute in Computer Science in France.

Set up in 1967 at Rocquencourt near Paris, INRIA is the French National Institute for Research in Computer Science and Control. It is a scientific and technological institute operating under the dual authority of the Ministry of Research and the Ministry of Industry.

The main missions of INRIA are: to undertake basic and applied research; to create experimental systems; to organize international scientific exchange; to ensure the transfer and dissemination of knowledge and expertise; to contribute to the effective implementation of research findings; to contribute to cooperative development programmes, especially through training; to carry out scientific evaluations; to contribute to standardization.

INRIA has maintained for many years a solid reputation in basic and applied research in the fields of computer science and control. Along with its international experience, INRIA has made itself available to industry in order to realize the best solutions possible and bring them to market fast. Therefore, INRIA is closely involved in the transfer of technology, either through partnership with industrial companies, or through its high-tech companies. One of its most noticeable achievements has been to support development and marketing activities for products from those high-tech companies.

INRIA participates in the project with people from 2 research groups: Mimosa and Lemme. The Mimosa group, to which Roberto Amadio and Davide Sangiorgi belong, has worked on models and calculi for mobility for the past 10 years, especially on types and co-inductive techniques. The group has participated in several projects, both at national and international level; these include the ESPRIT BRA projects CONCUR I and CONCUR II, CONFER, the HCM network EXPRESS, the working group CONFER II. In the past 2 years the group has moved its interests more specifically towards (mobile) distributed systems, in particular logics, types, and security protocols. The group presently participates in (and coordinates) the French "Action Concertee incitative" Vernam, on automatic verification of cryptographic protocols.

The Lemme group, to which Gilles Barthe belongs and of which Daniel Hirschkoff is an external collaborator, has a strong experience on type systems, program analysis and program transformation, proof assistants (theorem provers). In the past few years the group is working on applications of types and theorem provers to security, in collaborations with industry (Gemplus and Trusted Logic; the latter is a start-up of INRIA). The group presently participates in (and coordinate) the French "Action de Recherche Coopérative" S-Java, on the theme of environments for verifying the security of Java platforms and Java programs, and in the European project VerifiCard, whose goal is to establish the correctness of the JavaCard platform and to develop tools for proving security properties of JavaCard applications.

The work the 3 above-mentioned projects (Vernam, S-Java,VerifiCard) is relevant for PROFUNDIS. The participation of people from these 3 projects is important; for instance, it will help monitoring the advances of other research groups. In comparison with the European project VerifiCard, PROFUNDIS more specifically targets techniques for automated verification, and is not necessarily aiming at JavaCard-based applications.

Davide Sangiorgi Researcher. Ph.D. 1992 at Edinburgh; Research fellow at Edinburgh 1993-1994; at INRIA Sophia since beginning 1995. His research activity has touched several topics related to the mobile processes, in particular π-calculus, including: type systems; proof techniques; the comparison with the lambda-calculus; models; co-inductive techniques; algebraic theory; expressiveness.

Roberto Amadio Professor of Computer Science at the "Université de Provence", in Marseille (France). PhD 1991 University of Pisa; Research Fellow of CNRS in Nancy and Nice between 1991 and 1996. Member INRIA project MIMOSA.

Research interests: Lambda-calculus, Domain Theory, Type Theory, Modelling and Verification of Distributed Systems, Automated Deduction.

Gilles Barthe Researcher at INRIA Sophia-Antipolis since October 1999. Ph. D. in 1993 at the University of Manchester. Previous affiliations: University of Nijmegen (NL) 1993-1995, CWI, Amsterdam (NL) 1995-1997, Chalmers University (S) 1997-1998, Minho University (1998-1999). His current research interests include lambda calculus, type systems, formal verification techniques for smartcards and security.

Ilaria Castellani Researcher at INRIA Sophia-Antipolis since 1987. She obtained a Master degree from Pisa University in 1981 and a PhD from Edinburgh University in 1988. Her research interests include: concurrency theory, non-interleaving semantics for process calculi, process calculi with localities, asynchronous calculi for mobile processes, secure information flow.

Daniel Hirschkoff Lecturer. Ph.D. 1999 at École Nationale des Ponts et Chaussées (Paris); Lecturer at École Normale Supérieure de Lyon since September 1999. Research interests: theorem prover formalizations of mobile processes, automated deduction, verification of mobile processes.

Other participants Charles Meyssonnier, Etienne Lozes, Vincent Vanackère.
Selected publications: [22, 25, 26, 27, 28, 13, 30, 37].

Dipartimento di Informatica, Università di Pisa - PISA (PISA)

The Dipartimento di Informatica of Pisa offers two levels of undergraduate curricula: a three years program ("Diploma Universitario in Informatica"), and a five years one ("Laurea in Informatica"). It also offers a post-graduate school in Computer Science ("Dottorato").

The Department has at present about 80 members (40 professors, 20 assistant professors and 20 administrative and technical staff members). About 35 post-graduate students are enrolled in the PhD courses.

The main areas of current research in the Department are: Algorithms and Data Structures, Computer architecture, Artificial Intelligence and Robotics, Databases and Information Retrieval, Computational Mathematics, Programming Languages, Software Methodology and Engineering. The Department was evaluated in 1999 by an international team of reviewers. The evaluation exercise and its results are described at the address `http://www.di.unipi.it/Evaluation`. The Department has a large research production and is involved either as partner or as coordinator in many national and international projects. The Department also carries out research programs in cooperation with several computer companies on medium and long term research themes (see the Annual Research Report at `http://www.di.unipi.it/arr99/`).

The Research Group on Models & Languages for Open Distributed Systems, Dipartimento di Informatica, University of Pisa, has a strong background in the areas of mobility and coordination.

In particular the Group worked on the π-calculus: a concurrent and distributed version of it was developed using graph rewriting techniques [58]; its operational definition was expressed in a restricted format, which automatically yields a reduction semantics [52]; and finite state verification was made possible for its finitary version [55]. In this line, we participated in ESPRIT project CONFER, and working group CONFER2, both dedicated to functional languages and calculi for concurrency and mobility.

The step from the π-calculus towards programming languages for mobility and coordination was taken recently with the definition [46] and initial implementation [38] of the language KLAIM (in collaboration with the university of Florence). Special attention was given to types for security and access control [49, 47, 48]. We participated in the ESPRIT working group COORDINA, which hosts most of the European community on the subject. Currently, the group participates (together with academic groups of Bologna, Florence and Milan) in the project "Network aware programming and Interoperability funded by Microsoft Research, Cambridge, UK.

History Dependent Automata (HD-automata) have been introduced in [56, 59], which are able to allocate and garbage collect names. An extended format able to represent HD-automata has been defined within the verification environment JACK developed in Pisa at IEI-CNR by Stefania Gnesi, and a tool able to translate from the π-calculus to HD-automata has been implemented [51]. Behavioural properties related to dynamic network connectivity, locality of resources and processes and causality among events can be translated to finite HD-automata and formally verified. A coalgebraic foundation of HD-automata has been introduced in [57].

Michele Boreale received a Laurea degree in Scienze dell'Informazione in 1991 from the University of Pisa, and Ph.D. degree in Computer Science in 1995 from the University "La Sapienza", Rome. He has been research associate at the Dipartimento di Scienze dell'Informazione of University "La Sapienza" from February 1996 to July 1999, when he moved to the Dipartimento di Sistemi e Informatica of the University of Florence. Boreale is interested in formal methods for specifying and verifying concurrent and reactive systems. His work aims at investigating expressive power and tractable proof methods for diverse notions of process calculi and behavioral equivalences. Emphasis is on those aspects that are critical to distributed systems, such as asynchrony, mobility and cryptography in communications.

Gianluigi Ferrari is associate professor of Computer Science at Pisa University. His research interests generally fall in the area of semantic theories for concurrent-distributed programming and specification languages. He is also interested in the design and development of programming languages for highly distributed networks and semantic-based verification environments. He has taken part to many EU and Italian MURST and CNR research projects, or by industries projects as "Network aware programming and Interoperability" funded by Microsoft Research, Cambridge, UK.

Stefania Gnesi graduated in computer science at the University of Pisa

in 1978. She is a researcher at IEI-CNR since 1984. She was a lecturer of Software Engineering at the University of Siena from 1994 to 2000. Currently she is lecturer of Software Engineering at the University of Florence. She was responsible for the development of the formal specification and verification environment JACK and served as coordinator for several CNR projects on Methods and Tools for Analysis, Verification and Validation of Safety-Critical and Mobile Systems. Gnesi's current research interests include methods and tools for the formal specification and verification of concurrent, distributed and mobile systems, and applications of model checking to safety-critical and mobile systems and to security protocols.

Ugo Montanari is full professor of computer science in Pisa since 1976. His interests include Semantics of Concurrency, Process Description and Object Oriented Languages, Constraint Programming, Graph Rewriting Systems, Coordination Models, Software Architectures, Algebraic and Categorical Models of Concurrency. Pioneering work in: picture recognition, graphics, graph grammars, heuristically guided search, networks of constraints, algebraic data types, logic unification and true concurrency. Presently site coordinator for Esprit Working Group APPLIGRAPH and recently for TMR Network GETGRATS and Esprit Working Groups CONFER II and COORDINA.

Marco Pistore graduated and got the PhD degree in 1998 [59] in computer science from the University of Pisa. Marco Pistore is a researcher at ITC-IRST since 1999. His current research interests are: formal methods and model checking; the application of model checking techniques to the verification of distributed multi-agent systems; and the integration of formal methods into the requirements engineering process. Marco Pistore is also involved in several industrial technology transfer projects aiming at the formal specification and verification of safety critical systems (e.g. railways systems, embedded systems).

Due to the number of senior researcher involved in the project, and to the need of frequent traveling to carry on the advanced research necessary to the success of the project, sufficient resources are needed for travel and subsistence, and have been exposed in form A4.

Recent selected publications: [46, 51, 47, 49, 45, 53, 50, 40, 39, 41].

References

[Aba97] M. Abadi. Secrecy by typing in security protocols. In M. Abadi and T. Ito, editors, *Proc. TACS '97*, volume 1281 of *Lecture Notes in Computer Science*. Springer Verlag, 1997.

[AG99] Abadi, M., Gordon, A. A Calculus for Cryptographic Protocols: The spi-calculus. Information and Computation, 148(1):1-70, Academic Press, 1999.

[CG98] L. Cardelli and A. D. Gordon. Mobile ambients. In Maurice Nivat, editor, *Proceedings of the First International Conference on Foundations of Software Science and Computation Structures (FoSSaCS '98), Held as Part of the Joint European Conferences on Theory and Practice of Software (ETAPS'98), (Lisbon, Portugal, March/April 1998)*, volume 1378 of *lncs*. sv, 1998.

[CH00] L. Cardelli and A. D. Gordon. Anytime, anywhere. modal logics for mobile ambients. In *Proceedings of the 27th ACM Symposium on Principles of Programming Languages*, pages 365–377, 2000.

[CG01] L. Cardelli and A. D. Gordon. Logical properties of name restriction. In *Typed Lambda Calculi and Applications*, To appear, 2001.

[CS00] G.L. Cattani and P. Sewell. Models for name-passing processes: Interleaving and causal (extended abstract). In *15th Annual Symposium on Logic in Computer Science*. IEEE Computer Society Press, 2000.

[CJM98] E. Clarke, S. Jha, and W. Marrero. Using state space exploration and a natural deduction style message derivation engine to verify security protocols. In *Proc. IFIP Working Conference on Programming Concepts and Methods (PROCOMET)*, 1998.

[CrWe00] K. Crary and S. Weirich. Resource bound certification. In *Proc. 27th POPL*, pages 184–198, N.Y., 2000. ACM Press.

[Dal98] S. Dal-Zilio. Concurrent objects in the blue calculus. Submitted, 1998.

[FPT99] M.P. Fiore, G.D. Plotkin and D. Turi. Abstract syntax and variable binding. In *14th Annual Symposium on Logic in Computer Science*. IEEE Computer Society Press, 1999.

[FGM00] R. Focardi, R. Gorrieri and F. Martinelli. Non Interference for the Analysis of Cryptographic Protocols. In Ugo Montanari, Jose Rolim and Emo Welzl, Eds., ICALP 2000, Springer LNCS 1853, pages 354–372.

[GP99] M.J. Gabbay and A.M. Pitts. A new approach to abstract syntax involving binders. In *14th Annual Symposium on Logic in Computer Science*. IEEE Computer Society Press, 1999.

[Gir87] J.-Y. Girard. Linear Logic. *Theoretical Computer Science*, 50:1-102, 1987.

[Gou00] J. Goubault-Larrecq. A method for automatic cryptographic protocol verification. In *Proc. FMPPTA, Springer-Verlag*, 2000.

[HM85] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *JACM*, 32(1):137–161, 1985.

[Hof99] M. Hofmann. Linear types and non-size-increasing polynomial time computation. In *14th Symposium on Logic in Computer Science (LICS'99)*, pages 464–473, Washington - Brussels - Tokyo, July 1999. IEEE.

[Hon93] K. Honda. Types for dyadic interaction. In E. Best, editor, *Proc. CONCUR '93*, volume 715 of *Lecture Notes in Computer Science*, pages 509–523. Springer Verlag, 1993.

[Hon00] K. Honda. Elementary structures in process theory (1): Sets with renaming. *Mathematical Structures in Computer Science*, 10:617-663, 2000.

[How96] D. J. Howe. Proving congruence of bisimulation in functional programming languages. *Information and Computation*, 124(2):103–112, 1996.

[HKMN99] H. Hüttel, J. Kleist, M. Merro, and U. Nestmann. Migration = cloning ; aliasing. To be presented at Sixth Workshop on Foundations of Object-Oriented Languages (FOOL 6), 1999.

[IgKo01] A. Igarashi and N. Kobayashi. A generic type system for the pi-calculus. In *Proc. 28th POPL*, pages 128–141, N.Y., 2001. ACM Press.

[KPT99] N. Kobayashi, B.C. Pierce, and D.N. Turner. Linearity and the pi-calculus. *TOPLAS*, 21(5):914–947, 1999. Preliminary summary appeared in Proceedings of POPL'96.

[Low96] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Proc. TACAS, Springer Lect. Notes in Comp. Sci. 1996*, 1996.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[MMS97] J. Mitchell, M. Mitchell, and U. Stern. Automated analysis of cryptographic protocols using murϕ. In *Proc. IEEE Symposium on Security and Privacy*, 1997.

[MiPl88] J. C. Mitchell and G. D. Plotkin. Abstract types have existential type. *ACM Trans. Prog. Lang. and Sys.*, 10(3):470–502, 1988.

[Mon99] D. Monniaux. Abstracting cryptographic protocols with tree automata. In *Proc. Static Analysis Symposium, Springer Lect. Notes in Comp. Sci.*, 1999

[MWCG98] G. Morrisett, D. Walker, K. Crary, and N. Glew. From system F to typed assembly language. In *Proc. 25th POPL*, pages 85–97, New York, NY, 1998. ACM.

[Nec97] G.C. Necula. Proof-carrying code. In *Proc. 24th POPL*. ACM Press, 1997.

[PG00] A.M. Pitts and M.J. Gabbay. A metalanguage for programming with bound names modulo renaming. In R. Backhouse and J.N. Oliveira, editors, *Mathematics of Program Construction, 5th International Conference, MPC2000*, volume 1837 of *Lecture Notes in Computer Science*. Springer, 2000.

[RaVa97] A. Ravara and V.T. Vasconcelos. Behavioural types for a calculus of concurrent objects. In *3rd International Euro-Par Conference*, volume 1300 of *Lecture Notes in Computer Science*, pages 554–561. Springer Verlag, 1997.

[Rey88] J. C. Reynolds. Preliminary design of the programming language Forsythe. Tech. rept. CMU-CS-88-159. Carnegie Mellon University., 1988.

[RE99] C. Röckl and J. Esparza. Proof-checking protocols using bisimulations. In J.C.M. Baeten and S. Mauw, editors, *Proc. CONCUR'99*, volume 1664 of *Lecture Notes in Computer Science*, pages 525–540. Springer, 1999.

[Sew98] Peter Sewell. Global/local subtyping and capability inference for a distributed π-calculus. In Kim G. Larsen, Sven Skyum, and Glynn Winskel, editors, *25th Colloquium on Automata, Languages and Programming (ICALP) (Aalborg, Denmark)*, volume 1443 of *LNCS*, pages 695–706. Springer, July 1998.

[SmVo98] G. Smith and D. Volpano. Secure information flow in a multi-threaded imperative language. In *Proc. 25th POPL*. ACM Press, 1998.

[SuKa98] E. Sumii and N. Kobayashi. A generalized deadlock-free process calculus. In U. Nestmann and B.C. Pierce, editors, *HLCL '98: High-Level Concurrent Languages*, volume 16.3 of *ENTCS*. Elsevier Science Publishers, 1998.

[Yos96] N. Yoshida. Graph types for monadic mobile processes. In *Proc. FST & TCS*, volume 1180 of *Lecture Notes in Computer Science*, pages 371–386. Springer Verlag, 1996. Full paper appeared as Technical Report, ECS-LFCS-96-350, 1996, Edinburgh.

.

References

* References of UU

[1] Thomas Arts, Mads Dam, Lars åke Fredlund, and Dilian Gurov. Verification of distributed Erlang programs. In Claude Kirchner and Hélène Kirchner, editors, *Proceedings of the 15th International Conference on Automated Deduction (CADE-98)*, volume 1421 of *LNAI*, pages 38–41, Berlin, July 5–10 1998. Springer.

[2] Rance Cleaveland, Joachim Parrow, Bernhard Steffen. The Concurrency Workbench: A Semantics Based Tool for the Verification of Concurrent Systems. ACM Transactions on Programming Languages and Systems 15:36-72 (1993).

[3] M. Dam. Model checking mobile processes. *Information and Computation*, 129(1):35–51, 25 August 1996.

[4] Mads Dam. Proving properties of dynamic process networks. *Journal of Information and Computation*, 140(2):95–114, February 1998.

[5] M. Dam: "Proof Systems for Pi-Calculus Logics". To appear. de Queiroz (ed.), "Logic for Concurrency and Synchronisation", Studies in Logic and Computation, Oxford Univ Press, 2001.

[6] M. Dam and P. Giambiagi. "Confidentiality for Mobile Code: The Case of a Simple Payment Protocol". Proc. 13th Computer Science Foundations Workshop, 2000.

[7] Dam, L.-a. Fredlund and D. Gurov. "Toward Parametric Verification of Open Distributed Systems." In Compositionality: The Significant Difference (H. Langmaack, A. Pnueli, W.-P. De Roever (eds.)), Springer-Verlag 1998.

[8] M. Dam and D. Gurov. "Mu-Calculus with Explicit Points and Approximations (Abstract)". Proc. Fixed Points in Computer Science, Paris, 2000.

[9] Cosimo Laneve and Björn Victor. Solos in Concert. In: Jiří Wiederman, Peter van Emde Boas and Mogens Nielsen, Eds., Proceedings of ICALP'99, Springer LNCS 1644, pp.513–523, 1999.

[10] Robin Milner, Joachim Parrow, David Walker. A Calculus of Mobile Processes - Part I and Part II. Information and Computation 100:1-77 (1992).

[11] R. Milner, J. Parrow, and D. Walker. Modal logics for mobile processes. *Theoretical Computer Science*, 114:149–171, 1993.

[12] Fredrik Orava, Joachim Parrow. An Algebraic Verification of a Mobile Network. Formal Aspects of Computing 4:497-543 (1992).

[13] J. Parrow and D. Sangiorgi. Algebraic theories for name-passing calculi. *Information and Computation*, 120(2):174–197, 1995.

[14] Joachim Parrow and Björn Victor. The Fusion Calculus: Expressiveness and Symmetry in Mobile Processes, Proceedings of LICS '98, IEEE Computer Society Press.

[15] B. Victor and F. Moller. The Mobility Workbench — a tool for the π-calculus. In D. Dill, editor, *CAV'94: Computer Aided Verification*, volume 818 of *Lecture Notes in Computer Science*, pages 428–440. Springer-Verlag, 1994.

[16] J.-L. Vivas and M. Dam. "From Higher-Order pi-Calculus to pi-Calculus in the Presence of Static Operators". Proc. CONCUR'98.

* References of FFCT

[17] L. Caires. *A model for declarative programming and specification with concurrency and mobility.* PhD thesis, Departamento de Informática, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa (`http://ctp.di.fct.unl.pt/~lcaires/tese/`), 1999.

[18] L. Caires and L. Monteiro. Verifiable and executable specifications of concurrent objects in $\mathcal{L}_\pi$. In C. Hankin, editor, *ESOP European Symposium on Programming Languages and Systems, ETAPS'98*, number 1381 in Lecture Notes in Computer Science, pages 42–56. Springer-Verlag, 1998.

[19] L. Monteiro. Semantic domains based on sets with families of equivalences. In B. Jacobs, L. Moss, H. Reichel, and J. Rutten, editors, Coalgebraic Methods in Computer Science (CMCS'98), *Electronic Notes in Theoretical Computer Science* 11 (1998), 73–106.

[20] L. Monteiro. Observation Systems. In B. Jacobs, L. Moss, H. Reichel, and J. Rutten, editors, Coalgebraic Methods in Computer Science (CMCS'2000), *Electronic Notes in Theoretical Computer Science* (2000).

* References of INRIA

[21] R. M. Amadio and M. Dam. Toward a modal theory of types for the π-calculus. In *Proc. FTRTFT'96*, volume 1135 of *lncs*. sv, 1996.

[22] R. Amadio and D. Lugiez. On the reachability problem in cryptographic protocols. In *Proc. CONCUR00, Springer Lect. Notes in Comp. Sci. 1877*, 2000. Also appeared as RR-INRIA 3915.

[23] R. Amadio and D. Lugiez and V. Vanackere. On the symbolic reduction of processes with cryptographic functions. Research Report INRIA-Sophia, March 2001.

[24] R. Amadio and S. Prasad. The game of the name in cryptographic tables. In *Proc. ASIAN99, Springer LNCS 1742*, 1999.

[25] G. Barthe, G. Dufay, L. Jakubiec, S. Melo de Sousa, and B. Serpette. A Formal Executable Semantics of the JavaCard Platform. In D. Sands, editor, *Proceedings of ESOP'01*, volume 2028 of *Lecture Notes in Computer Science*, pages 302–319. Springer Verlag, 2001.

[26] G. Barthe and B. Serpette. Partial evaluation and non-interference for object calculi. In A. Middeldorp and T. Sato, editors, *Proceedings of FLOPS'99*, volume 1722 of *Lecture Notes in Computer Science*, pages 53–67. Springer Verlag, 1999.

[27] Daniel Hirschkoff. A full formalization of π-calculus theory in the Calculus of Constructions. In E. L. Gunter and A. Felty, editors, *Theorem Proving in Higher-Order Logic: 10th International Conference, TPHOLs'97*, volume 1275 of *LNCS*, pages 153–169, 1997.

[28] D. Hirschkoff. On the benefits of using the up-to techniques for bisimulation verification. In W. Rance Cleaveland, editor, *Proceedings TACAS '99*, volume 1579 of *Lecture Notes in Computer Science*, pages 286–299. , 1999.

[29] J. Kleist and D. Sangiorgi. Imperative objects and mobile processes. In *Proc. IFIP Working Conference on Programming Concepts and Methods (PROCOMET'98)*. North-Holland, 1998.

[30] B. Pierce and D. Sangiorgi. Typing and subtyping for mobile processes. *Journal of Mathematical Structures in Computer Science*, 6(5):409–454, 1996. An extended abstract in *Proc. LICS 93*, IEEE Computer Society Press.

[31] B. Pierce and D. Sangiorgi. Behavioral equivalence in the polymorphic pi-calculus. *Journal of the ACM*, 47(3):531–584, 2000.

[32] D. Sangiorgi. Typed π-calculus at work: a proof of Jones's parallelisation transformation on concurrent objects. Proc. Fourth Workshop on Foundations of Object-Oriented Languages (FOOL 4)., 1997.

[33] D. Sangiorgi. On the bisimulation proof method. *Journal of Mathematical Structures in Computer Science*, 8:447–479, 1998.

[34] D. Sangiorgi. The name discipline of uniform receptiveness. *Theoretical Computer Science*, 221:457–493, 1999.

[35] D. Sangiorgi. Extensionality and intensionality of the ambient logic. In *Proc. 28th POPL*. ACM Press, 2001.

[36] D. Sangiorgi and R. Milner. The problem of "Weak Bisimulation up to". In W.R. Cleveland, editor, *Proc. CONCUR '92*, volume 630 of *Lecture Notes in Computer Science*, pages 32–46. Springer Verlag, 1992.

[37] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes.* Cambridge University Press, 2001, To appear.

* References of PISA

[38] L. Bettini, R. De Nicola, G. Ferrari, R. Pugliese, Interactive Mobile Agents in XKLAIM, Int. Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprise, IEEE Press, 1998.

[39] M. Boreale. Symbolic trace analysis of cryptographic protocols. Accepted to ICALP'01, LNCS, Springer.

[40] M. Boreale, R. De Nicola, R. Pugliese. Basic Observable for Processes. Information and Computation 149, 1999.

[41] M. Boreale, R. De Nicola, R. Pugliese. Proof Techniques for Cryptographic Processes. To appear in SIAM Journal on Computing.

[42] M. Boreale and D. Sangiorgi. Bisimulation in name-passing calculi without matching. In *Proc. 13th LICS Conf.* IEEE Computer Society Press, 1998.

[43] M. Boreale and D. Sangiorgi. A fully abstract semantics for causality in the π-calculus. *Acta Informatica*, 35:353–400, 1998.

[44] M. Boreale and D. Sangiorgi. Some congruence properties for π-calculus bisimilarities. *Theoretical Computer Science*, 198:159–176, 1998.

[45] De Francesco, A. Fantechi, S. Gnesi, P.Inverardi, "Finite Approximations for Model Checking non-finite state processes", Computer Journal, vol 44, no 2, Oxford University Press, to appear in 2001.

[46] R. De Nicola, G. Ferrari, R. Pugliese. KLAIM: A Kernel Language for Agent Interaction and Mobility. IEEE Transactions on Software Engineering, Vol 24 (5), 1998.

[47] R. De Nicola, G. Ferrari, R. Pugliese Types as Specification of Access Policies In *Secure Internet Programming: Security Issues for Distributed and Mobile Objects*, LNCS State-Of-The-Art-Survey (J. Vitek and C. Jensen Eds.), LNCS 1603, pp 117-146, 1999.

[48] R. De Nicola, G. Ferrari, R. Pugliese, Programming Access Control: The KLAIM Experience, In Proc. *CONCUR'2000*, LNCS, 2000

[49] R. De Nicola, G. Ferrari, R. Pugliese, B. Venneri Types for Access Control, *Theoretical Computer Science*, 240(1):215-254, Elsevier Science, 2000.

[50] Fantechi, S. Gnesi, F. Mazzanti, R. Pugliese, E. Tronci "A Symbolic Model Checker for ACTL", Applied Formal Methods – FM-Trends 98, International workshop, LNCS 1641, Springer - Verlag, 1999.

[51] G. Ferrari, S. Gnesi, U. Montanari, M. Pistore, G. Ristori, Verifying Mobile Processes in the HAL Environment. *Computer Aided Verification* (CAV'98), LNCS 1427, 1998.

[52] Ferrari, G., Montanari, U. and Quaglia, P. A π-calculus with Explicit Substitutions. Theoretical Computer Science Vol. 168, 1, Nov. 1996, pp.53-103.

[53] Gnesi, G.Ristori. "A Model Checking Algorithm for π-calculus agents. In Advances in Temporal Logic, H. Barringer, M. Fisher, D. Gabbay, G. Gough eds., Applied Logic Series, Vol. 16, Kluwer Academic Publishers, 2000, pp. 339-358.

[54] F. Levi and D. Sangiorgi. Controlling interference in ambients. In *Proc. 27th POPL*. ACM Press, 2000.

[55] Montanari, U. and Pistore, M. Checking Bisimilarity for Finitary pi-calculus. In: Insup Lee, Scott A. Smolka, Eds., CONCUR'95: Concurrency Theory, Springer LNCS 962, pp. 42-56.

[56] Montanari, U. and Pistore, M. An Introduction to History Dependent Automata. In: Andrew Gordon, Andrew Pitts and Carolyn Talcott, Eds, Second Workshop on Higher-Order Operational Techniques in Semantics (HOOTS II), ENTCS, Vol. 10, 1998.

[57] U. Montanari and M. Pistore. π-calculus, structured coalgebras and minimal HD-automata. In M. Nielsen and B. Rovan, editors, *Mathematical Foundations of Computer Science 2000*, volume 1893 of *Lecture Notes in Computer Science*. Springer, 2000.

[58] Montanari, U., Pistore, M. and Rossi, F. Modeling Concurrent, Mobile and Coordinated Systems via Graph Transformations In: G. Rozenberg, Ed., Handbook of Graph Grammars and Computing by Graph Transformation, Vol.3: Concurrency, Parallellism, and Distribution, to appear.

[59] M. Pistore. *History Dependent Automata*. PhD. Thesis TD-5/99, Università di Pisa, Dipartimento di Informatica, 1999.

[60] M. Pistore and D. Sangiorgi. A partition refinement algorithm for the π-calculus. In *Proc. CAV'96*, volume 1102 of *Lecture Notes in Computer Science*, pages 38–49. Springer Verlag, 1996. Extended version to appear in Information and Computation.