IN PARTNERSHIP WITH:
Université de Bologne (Italie)

Activity Report 2015

Project-Team FOCUS

Foundations of Component-based Ubiquitous Systems

IN COLLABORATION WITH: Dipartimento di Informatica - Scienza e Ingegneria (DISI), Universita' di Bologna

RESEARCH CENTER
Sophia Antipolis - Méditerranée

THEME
Distributed programming and Software engineering

Table of contents

Project-Team FOCUS

Creation of the Project-Team: 2010 January 01

Keywords:

Computer Science and Digital Science:

1. - Architectures, systems and networks
1.3. - Distributed Systems
1.4. - Ubiquitous Systems
2. - Software
2.1. - Programming Languages
2.1.1. - Semantics of programming languages
2.1.5. - Constraint programming
2.1.6. - Concurrent programming
2.1.7. - Distributed programming

Other Research Topics and Application Domains:

6.1. - Software industry
6.3. - Network functions
6.4. - Internet of things
9.4.1. - Computer science

1. Members

Research Scientist

Elena Giachino [University Bologna, Envisage project]

Faculty Members

Davide Sangiorgi [Team leader,University Bologna, Professor, HdR]
Mario Bravetti [University Bologna, Associate Professor]
Ugo Dal Lago [University Bologna, Associate Professor]
Maurizio Gabbrielli [University Bologna, Professor]
Ivan Lanese [University Bologna, Lecturer]
Cosimo Laneve [University Bologna, Professor]
Simone Martini [University Bologna, Professor, HdR]
Gianluigi Zavattaro [University Bologna, Associate Professor]

Engineer

Jacopo Mauro [Univ. Paris VII, granted by ANR AEOLUS project, until Aug 2015]

PhD Students

Michele Alberti [Institut de Mathématiques de Luminy Marseille, until Mar 2015]
Roberto Amadini [University Bologna]
Alberto Cappai [University Bologna]
Francesco Gavazzo [University Bologna, from Oct 2015]
Saverio Giallorenzo [University Bologna]
Jean-Marie Madiot [ENS Lyon, until Apr 2015]
Alessandro Rioli [University Bologna]
Marco Solieri [Université Paris 13]
Valeria Vignudelli [University Bologna]

Post-Doctoral Fellows

Martin Avanzini [University Bologna, from May 2014]
Alexis Bernadet [Inria, until Oct 2015]
Flavien Breuvart [Inria, from Nov 2015]
Michael Lienhardt [University Bologna, Envisage project, until May 2015]

Administrative Assistant

Christine Riehl

Others

Giulio Pellitta [University Bologna]
Daniel Hirschkoff [External collaborator, ENS Lyon]
Claudio Guidi [External collaborator, italianaSoftware s.r.l.]
Fabrizio Montesi [External collaborator, IT University of Copenhagen]

2. Overall Objectives

2.1. Overall Objectives

Ubiquitous Computing refers to the situation in which computing facilities are embedded or integrated into everyday objects and activities. Networks are large-scale, including both hardware devices and software agents. The systems are highly mobile and dynamic: programs or devices may move and often execute in networks owned and operated by others; new devices or software pieces may be added; the operating environment or the software requirements may change. The systems are also heterogeneous and open: the pieces that form a system may be quite different from each other, built by different people or industries, even using different infrastructures or programming languages; the constituents of a system only have a partial knowledge of the overall system, and may only know, or be aware of, a subset of the entities that operate on the system.

A prominent recent phenomenon in Computer Science is the emerging of interaction and communication as key architectural and programming concepts. This is especially visible in ubiquitous systems. Complex distributed systems are being thought of and designed as structured composition of computational units, usually referred to as *components*. These components are supposed to interact with each other and such interactions are supposed to be orchestrated into conversations and dialogues. In the remainder, we will write *CBUS* for Component-Based Ubiquitous Systems.

In CBUS, the systems are complex. In the same way as for complex systems in other disciplines, such as physics, economics, biology, so in CBUS theories are needed that allow us to understand the systems, design or program them, analyse them.

Focus investigates the semantic foundations for CBUS. The foundations are intended as instrumental to formalizing and verifying important computational properties of the systems, as well as to proposing linguistic constructs for them. Prototypes are developed to test the implementability and usability of the models and the techniques. Throughout our work, ‘interaction’ and ’component’ are central concepts.

The members of the project have a solid experience in algebraic and logical models of computation, and related techniques, and this is the basis for our study of ubiquitous systems. The use of foundational models inevitably leads to opportunities for developing the foundational models themselves, with particular interest for issues of expressiveness and for the transplant of concepts or techniques from a model to another one.

3. Research Program

3.1. Models

The objective of Focus is to develop concepts, techniques, and possibly also tools, that may contribute to the analysis and synthesis of CBUS. Fundamental to these activities is *modeling*. Therefore designing, developing and studying computational models appropriate for CBUS is a central activity of the project. The models are used to formalize and verify important computational properties of the systems, as well as to propose new linguistic constructs.

The models we study are in the process calculi (e.g., the π-calculus) and λ-calculus tradition. Such models, with their emphasis on algebra, well address compositionality—a central property in our approach to problems. Accordingly, the techniques we employ are mainly operational techniques based on notions of behavioral equivalence, and techniques based on algebra, mathematical logics, and type theory.

4. Application Domains

4.1. Ubiquitous Systems

The main application domain for Focus are ubiquitous systems, broadly systems whose distinctive features are: mobility, high dynamicity, heterogeneity, variable availability (the availability of services offered by the constituent parts of a system may fluctuate, and similarly the guarantees offered by single components may not be the same all the time), open-endedness, complexity (the systems are made by a large number of components, with sophisticated architectural structures). In Focus we are particularly interested in the following aspects.

- *Linguistic primitives* for programming dialogues among components.
- *Contracts* expressing the functionalities offered by components.
- *Adaptability and evolvability* of the behaviour of components.
- *Verification* of properties of component systems.
- Bounds on component *resource consumption* (e.g., time and space consumed).

4.2. Service Oriented Computing and Cloud Computing

Today the component-based methodology often refers to Service Oriented Computing. This is a specialized form of component-based approach. According to W3C, a service-oriented architecture is “a set of components which can be invoked, and whose interface descriptions can be published and discovered”. In the early days of Service Oriented Computing, the term services was strictly related to that of Web Services. Nowadays, it has a much broader meaning as exemplified by the XaaS (everything as a service) paradigm: based on modern virtualization technologies, Cloud computing offers the possibility to build sophisticated service systems on virtualized infrastructures accessible from everywhere and from any kind of computing device. Such infrastructures are usually examples of sophisticated service oriented architectures that, differently from traditional service systems, should also be capable to elastically adapt on demand to the user requests.

5. Highlights of the Year

5.1. Highlights of the Year

5.1.1. Awards

- Ugo Dal Lago won the 2015 award of ‘Best Young Researcher in Theoretical Computer Science’ given by the Italian Chapter of the EATCS (European Association for Theoretical Computer Science).

- Ornela Dardha, former Focus PhD student, now at Glasgow University, won the award for 'Best Italian 2015 PhD Thesis in Theoretical Computer Science' given by the Italian Chapter of the EATCS.
- The Focus constraint programming solver, called `sunny-cp`, won the '2015 MiniZinc Challenge', http://www.minizinc.org/challenge.html, an annual competition of constraint programming solvers in the open category (the most challenging), featuring all the most efficient solvers in the world.
- Fabrizio Montesi, external collaborator in Focus, won the EAPLS (European Association for Programming Languages and Systems) 'Best PhD Dissertation Award 2014'.

BEST PAPER AWARD:

[44]

D. HIRSCHKOFF, J.-M. MADIOT, X. XU. *A behavioural theory for a π-calculus with preorders*, in "Proceedings of 6th IPM International Conference on Fundamentals of Software Engineering (FSEN 2015)", Teheran, Iran, LNCS, Springer Verlag, April 2015, This paper obtained the best paper award at the conference FSEN'2015. A revised version of this paper has been published in JLAMP (Journal of Logical and Algebraic Methods in Programming), Volume 84, Issue 6, November 2015, Pages 806–825. 2014, https://hal.archives-ouvertes.fr/hal-01246094

6. New Software and Platforms

6.1. AIOCJ

SCIENTIFIC DESCRIPTION

AIOCJ is an open-source choreography programming language for developing adaptive systems. It allows one to program adaptive distributed systems based on message passing. AIOCJ comes as a plugin for Eclipse, AIOCJ-ecl, allowing to edit descriptions of distributed systems as adaptive interaction-oriented choreographies (AIOC). From interaction-oriented choreographies the description of single participants can be automatically derived. Adaptation is specified by rules allowing to replace predetermined parts of the AIOC with a new behaviour. A suitable protocol ensures that all the participants are updated in a coordinated way. As a result, the distributed system follows the specification given by the AIOC under all changing sets of adaptation rules and environment conditions. In particular, the system is always deadlock free. AIOCJ can interact with external services, seen as functions, by specifying their URL and the protocol they support (HTTP, SOAP, ...). Deadlock-freedom guarantees of the application are preserved provided that those services do not block.

- Contact: Saverio Giallorenzo
- URL: http://www.cs.unibo.it/projects/jolie/aiocj.html

6.2. DF4ABS

Deadlock Framework for ABS

SCIENTIFIC DESCRIPTION

We have prototyped a framework for statically detecting deadlocks in a concurrent object-oriented language with asynchronous method calls and cooperative scheduling of method activations (the language is ABS, which has been developed in the EU project HATS and currently extended with primitives for cloud-computing in the EU project ENVISAGE. ABS is very similar to ASP, developed by the OASIS team.). Since this language features recursion and dynamic resource creation, deadlock detection is extremely complex and state-of-the-art solutions either give imprecise answers or do not scale. In order to augment precision and scalability we propose a modular framework that allows several techniques to be combined. The basic component of the framework is a front-end inference algorithm that extracts abstract behavioural descriptions of methods that retain resource dependency information. Then these behavioural descriptions are analysed by a back-end that uses a fix-point technique to derive in a deterministic way the deadlock information.

- Contact: Elena Giachino
- URL: http://df4abs.nws.cs.unibo.it/

6.3. HoCA

Higher-Order Complexity Analysis
SCIENTIFIC DESCRIPTION

Over the last decade, various tools for the static analysis of resource properties of programs have emerged. In particular, the rewriting community has recently developed several tools for the time complexity analysis of term rewrite systems. These tools have matured and are nowadays able to treat non-trivial programs, in a fully automatic setting. However, none of these automatic complexity analysers can deal with higher-order functions, a pervasive feature of functional programs.

Our tool HoCA (Higher-Order Complexity Analyser) overcomes this limitation by translating higher-order programs – in the form of side-effect free OCaml programs – into equivalent first-order rewrite systems. At the heart of our tool lies Reynold's defunctionalization technique. Defunctionalization however is not enough. Resulting programs have a recursive structure too complicated to be analysed automatically in all but trivial cases. To overcome this issue, HoCA integrates a handful of well established program transformation techniques, noteworthy dead-code elimination, inlining, instantiation and uncurrying. All these techniques have been specifically suited to the methods integrated in modern first-order complexity analysers. Of course, the complete transformation pipeline underlying our tool is not only proven semantically correct, but also to reflect the runtime behavior. This way, a complexity bound on the resulting first-order program can be relayed back reliably to the higher-order program of interest.
A detailed description of HoCA is available on http://arxiv.org/abs/1506.05043

- Contact: Ugo Dal Lago
- URL: http://cbr.uibk.ac.at/tools/hoca/

6.4. JOLIE

Java Orchestration Language Interpreter Engine
SCIENTIFIC DESCRIPTION

Jolie is a service-oriented programming language. Jolie can be used to program services that interact over the Internet using different communication protocols.

Differently from other Web Services programming languages such as WS-BPEL, Jolie is based on a user-friendly C/Java-like syntax (more readable than the verbose XML syntax of WS-BPEL) and, moreover, the language is equipped with a formal operational semantics. This language is used for the *proof of concepts* developed around Focus activities. For instance, contract theories can be exploited for checking the conformance of a Jolie program with respect to a given contract.
Developments in 2015: Jolie has transitioned from version 1.1 to version 1.4.1. The releases are the result of more than 400 commits with hundreds of bug fixes and enhancements. Highlights include: a new web site and documentation, a new pre-compiled installer, 2 new IDEs as plugins for the editors Sublime Text and Atom, a transition from SouceForge to GitHub, introduction of new behavioural and architectural constructs, structured support for the development of REST applications, introduction of the construct of internal services: embedded Jolie services defined directly within the embedder program (internal services offer a convenient way of reusing code as in procedural programming, without breaking the principle that such code should be easily exported to an external microservice), increased compliance with protocol standards (foremost HTTP, SSL), enhanced stability and performances. Moreover, 2015 has seen the development of Jolie Redeployment Optimiser (JRO), a tool for the automatic and optimised deployment of microservices written in Jolie. JRO uses Zephyrus, a state-of-the-art tool that automatically generates configurations starting from partial and abstract descriptions of the target application. Given the output configuration from Zephyrus, JRO interacts with Jolie Enterprise, an administrative tool for the deployment of Jolie services on remote nodes, to deploy the wanted architecture.

- Contact: Fabrizio Montesi, Saverio Giallorenzo
- URL: http://www.jolie-lang.org/

6.5. SRA

Static Resource Analyzer for ABS

SCIENTIFIC DESCRIPTION

We prototype a static analysis technique that computes upper bounds of virtual machine usages in a concurrent language with explicit acquire and release operations of virtual machines. In our language it is possible to delegate other (ad-hoc or third party) concurrent code to release virtual machines (by passing them as arguments of invocations, a feature that is used by Amazon Elastic Cloud Computing or by the Docker FiWare). Our technique is modular and consists of (i) a type system associating programs with behavioural descriptions that record relevant information for resource usage (creations, releases, and concurrent operations), (ii) a translation function that takes behavioural types and returns cost equations, and (iii) an automatic off-the-shelf solver for the cost equations.

- Contact: Elena Giachino
- URL: http://sra.cs.unibo.it/

6.6. SUNNY-CP

SCIENTIFIC DESCRIPTION

Within the Constraint Programming paradigm, a portfolio solver combines different constraint solvers in order to create a globally better solver. Sunny-cp is a parallel parallel portfolio solver capable of solving Constraint (Satisfaction/Optimization) Problems defined in the MiniZinc language. It essentially implements the SUNNY algorithm introduced in the team. Sunny-cp is built on top of state-of-the-art constraint solvers, including: Choco, Chuffed, CPX, G12/LazyFD, G12/FD, G12/Gurobi, G12/CBC, Gecode, HaifaCSP, iZplus, MinisatID, Opturion, OR-Tools

SUNNY-CP is a portfolio solver for solving both Constraint Satisfaction Problems and Constraint Optimization Problems. The goal of SUNNY-CP is to provide a flexible, configurable, and usable CP portfolio solver that can be set up and executed just like a regular individual CP solver.

- Contact: Roberto Amadini
- URL: https://github.com/CP-Unibo/sunny-cp

6.7. Blender

SCIENTIFIC DESCRIPTION

The various tools developed in the Aeolus project (Zephyrus, Metis, Armonic) have been combined in this software which represents an integrated solution for the declarative specification of cloud applications, and its subsequent automatic deployment on an OpenStack cloud system. In particular, a web-based interface is used to specify the basic software artifacts to include in the application, indicate their level of replication, and specify co-installability conflicts (i.e. when two components cannot be installed on the same virtual machines). The tool Zephyrus is then used to synthesize the final architecture of the application, the tool Metis indicates the plan of configuration actions, and the Armonic platform provides the library of components and the low-level scripts to actually install and configure the entire application.

- Partners: IRILL - Mandriva
- Contact: Gianluigi Zavattaro
- URL: https://github.com/aeolus-project/blender

7. New Results

7.1. Service-oriented computing

Participants: Maurizio Gabbrielli, Elena Giachino, Saverio Giallorenzo, Claudio Guidi, Mario Bravetti, Cosimo Laneve, Ivan Lanese, Michael Lienhardt, Jacopo Mauro, Fabrizio Montesi, Gianluigi Zavattaro.

7.1.1. Orchestrations

Orchestration models and languages in the context of Service-Oriented Architectures (SOA) are used to describe the composition of services focusing on their interactions. Concrete web services are connected to abstract service definitions for the aim of service discovery. In [16] we study a natural notion of compliance between clients and services in terms of their bpel (abstract) descriptions. The induced preorder shows interesting connections with the must preorder and has normal form representatives that are parallel-free finite-state activities, called contracts. Moreover, in [22] we focus on advancements of the orchestration language Jolie aiming at the development of dynamically adaptable orchestrated systems.

7.1.2. Choreographies

Choreographies are high-level descriptions of distributed interacting systems featuring as basic unit a communication between two participants. A main feature of choreographies is that they ensure deadlock-freedom by construction. From a choreography one can automatically derive a description of the behaviour of each participant using a notion of projection. Choreographies can be used both at the level of types (multiparty session types) or as a programming language. In [19] we surveyed our results about verification of adaptable processes, focusing in particular on distributed adaptability, where a process can update part of a protocol (specified by a choreography) by performing dynamic distributed updates over a set of protocol participants. In [14] we illustrate our approach to develop and verify distributed, adaptive software systems. The cornerstone of our framework is the use of choreography languages, which allow us to obtain correctness by construction. Moreover, in [36] we present DIOC, a language for programming distributed applications that are free from deadlocks and races by construction. A DIOC program describes a whole distributed application as a unique entity (choreography). DIOC allows the programmer to specify which parts of the application can be updated. At runtime, these parts may be replaced by new DIOC fragments from outside the application. DIOC programs are compiled, generating code for each site, in a lower-level language called DPOC. As a consequence DPOC applications are free from communication deadlocks and races, even in presence of runtime updates.

7.2. Models for reliability

Participants: Elena Giachino, Ivan Lanese.

7.2.1. Reversibility

We have continued the study of causal-consistent reversibility started in the past years. In [42] we defined the causal-consistent reversible semantics (both controlled and uncontrolled) of muKlaim, a formal coordination language based on distributed tuple spaces, by adapting the approach developed for message passing calculi in the past years. A major novelty is that the muKlaim read primitive allows two processes to access a shared resource independently, giving rise to a causality structure which is not found in message passing calculi.

In [31] we studied the issue of compliance of a client w.r.t. a server in a reversible setting using behavioural contracts. In particular, when an agreement cannot be reached, the client and the server can synchronously rollback to the last point of choice, looking for alternatives. As a main result, we showed that compliance is decidable even for recursive contracts.

7.3. Cloud Computing and Deployment

Participants: Elena Giachino, Saverio Giallorenzo, Claudio Guidi, Cosimo Laneve, Michael Lienhardt, Jacopo Mauro, Gianluigi Zavattaro.

7.3.1. Cloud application deployment

Configuration and management of applications in the cloud is a complex task that requires advanced methodologies and tools. A foundational study of the problem has been carried out in [21] where we have identified the critical tasks to be solved, quantified their computational complexity, and proposed simplifications to the problem with the idea of limiting the computational complexity at the cost of having approximated (but acceptable, in most cases) solutions. Our attention has been dedicated to the implementation of a tool for the efficient solution of one of these tasks, namely, the automatic planning of the management actions needed to properly configure a cloud application [17]. This tool, called Metis, has been already exploited in the realization of an integrated platform for the automatic deployment of the cloud application called Blender [39] as well as in the context of the ABS modeling language [37] in order to be able to support the automatic reasoning about deployment costs already during the early phases of application design and development. We have also performed a foundational study of the problem of reconfiguring an application instead of deploying it from scratch. Our foundational study allowed us, on the one hand, to quantify the computational complexity of the problem (PSpace-Complete) and, on the other hand, to precisely identify the source of such complexity (the presence of legacy components that cannot be re-deployed from scratch).

7.3.2. Cloud resource management

One of the main challenges in the management of cloud applications is the quantification of the computing resources needed by the applications to be deployed. More precisely, it is important to quantify upper bounds to the number of needed computing resources in order to either previously acquire them or have a precise quantification of the costs for executing an application. In [40] a static analysis technique is proposed that computes upper bounds of virtual machine usages in a concurrent language with explicit acquire and release operations of virtual machines. See the section on deadlock analysis for more details.

7.4. Resource Control and Probabilities

Participants: Michele Alberti, Martin Avanzini, Flavien Breuvart, Alberto Cappai, Ugo Dal Lago, Simone Martini, Giulio Pellitta, Alessandro Rioli, Davide Sangiorgi, Marco Solieri, Valeria Vignudelli.

7.4.1. Resource Control

7.4.1.1. Time Complexity Analysis of Concurrent and Higher-Order Functional Programs

We have extensively studied the problem of automatically analysing the complexity of programs. We first of all studied the problem for concurrent object-oriented programs [41]. To determine this complexity we have used intermediate abstract descriptions that record relevant information for the time analysis, called behavioural types. Behavioural types are then translated into so-called cost equations, making parallelism explicit. Cost equations are finally fed into an automatic off-the-shelf solver for obtaining the actual time complexity. The same problem has been also analysed when the underlying program is functional [29]. We showed how the complexity of higher-order functional programs can be analysed automatically by applying program transformations to a defunctionalized version of them, and feeding the result to existing tools for the complexity analysis of first-order term rewrite systems. This is done while carefully analysing complexity preservation and reflection of the employed transformations such that the complexity of the obtained term rewrite system reflects on the complexity of the initial program. This approach turns out to work well in practice, in particular since off-the-shelf complexity tool for first-order rewrite systems matured to a state where they are both fast and powerful. However, the implementation of such tools is quite sophisticated. To ensure correctness of the obtained complexity bounds, we extended CeTA, a certified proof checker for rewrite tools, with the formalisation of various complexity techniques underlying state-of-the-art complexity tools [30]. This way, we detected conflicts in theoretical results as well as bugs in existing complexity provers.

7.4.1.2. Function Algebras and Implicit Complexity

A fundamental result about ramified recurrence, one of the earliest systems in implicit complexity, has been proved [28]. This has been obtained through a careful analysis on how the adoption of an evaluation mechanism with sharing and memoization impacts the class of functions which can be computed in polynomial time. We have first shown how a natural cost model in which lookup for an already computed result has no cost is indeed invariant. As a corollary, we have then proved that the most general notion of ramified recurrence is sound for polynomial time.

7.4.1.3. Geometry of Interaction

We see the the geometry of interaction as a foundational framework in which the efficiency of higher-order computation can be analyzed. This has produced some very interesting results, also stimulated by the bilateral Inria project CRECOGI, which started this year. We have first of all studied the geometry of interaction of the resource lambda-calculus, a model of linear and nondeterministic functional languages. In a strictly typed restriction of the resource lambda-calculus, we have studied the notion of path persistence, and defined a geometry of interaction that characterises it [18]. Furthermore, we have carried out our work on multitoken machines, started in 2014. More specifically, we have studied multitoken interaction machines in the context of a very expressive linear logical system with exponentials, fixpoints and synchronization [34]. On the one hand, we have proved that interaction is guaranteed to be deadlock-free. On the other hand, the resulting logical system has been proved to be powerful enough to embed PCF and to adequately model its behaviour, both when call-by-name and when call-by-value evaluation are considered.

7.4.2. Probabilistic Models

7.4.2.1. Applicative Bisimilarity

Notions of equivalences for probabilistic programming languages have been studied and analysed, together with their relationships with context equivalence. More specifically, we have studied how applicative bisimilarity behaves when instantiated on a call-by-value probabilistic lambda-calculus, endowed with Plotkin's parallel disjunction operator [20]. We have proved that congruence and coincidence with the corresponding context relation hold for both bisimilarity and similarity, the latter known to be impossible in sequential languages. We have also shown that applicative bisimilarity works well when the underlying language of programs takes the form of a linear lambda-calculus extended with quantum data [35]. The main results are proofs of soundness for the obtained notion of equivalence.

7.4.2.2. From Equivalences to Metrics

The presence of probabilistic (thus quantitative) notions of observation makes equivalence relations too coarse-grained as ways to compare programs. This opens the way to metrics in which, indeed, not all non-equivalent programs are at the same distance. We have studied the problem of evaluating the distance between affine lambda-terms [33]. A natural generalisation of context equivalence has been shown to be characterised by a notion of trace distance, and to be bounded from above by a coinductively defined distance based on the Kantorovich metric on distributions. A different, again fully-abstract, tuple-based notion of trace distance has been shown to be able to handle nontrivial examples. A similar thing has been done in a calculus for probabilistic polynomial time computation [32], thus paving the way towards getting effective proof methodologies for computational indistinguishability, a key notion in modern cryptography.

7.5. Verification techniques for extensional properties

Participants: Daniel Hirschkoff, Elena Giachino, Michael Lienhardt, Cosimo Laneve, Jean-Marie Madiot, Davide Sangiorgi.

Extensional properties are those properties that constrain the behavioural descriptions of a system (i.e., how a system looks like from the outside). Examples of such properties include classical functional correctness, deadlock freedom and resource usage. Related to techniques for extensional properties are the issues of decidability (the problem of establishing whether certain properties are computationally feasible).

7.5.1. Static analysis of deadlock freedom and resource usage

Deadlock detection in concurrent programs that create networks with an arbitrary number of nodes is extremely complex and solutions either give imprecise answers or do not scale. To enable the analysis of such programs, we have studied an algorithm for detecting deadlocks in a basic concurrent object-oriented language. The algorithm (i) associates behavioural types, called *lam*, to programs by means of a type inference system and (ii) uses an ad-hoc verification technique highlighting circular dependencies in lam [15]. The algorithm has been prototyped and has been extended to a full-fledged programming language, called `ABS`.

A technique similar to [15] has been used for computing upper bounds of resource usages in [40]. In particular, the metaphor in this paper has been virtual machines usage in a concurrent language with explicit acquire and release operations. The problematic issue in such languages is when the release is delegated to other (ad-hoc or third party) concurrent codes (by passing them as arguments of invocations) – a feature that is currently used in `Amazon Elastic Cloud Computing` or in the `Docker FiWare`. As for deadlock analysis, the technique is modular and consists of (i) a type system associating programs with behavioural types that records relevant information for resource usage (creations, releases, and concurrent operations), (ii) a translation function that takes behavioural types and returns cost equations, and (iii) an automatic off-the-shelf solver for the cost equations. A soundness proof of the type system establishes the correctness of the above technique with respect to the cost equations. The technique has also been experimentally evaluated and the experiments show that it allows us to derive bounds for programs that are better than other techniques, such as those based on amortized analysis.

Another technique for enforcing program correctness is the one used in [36], [14], where the programming of distributed applications is guaranteed to be free from communication deadlocks and races by means of *choreographies*. Choreographies are behavioural types which allow one to obtain correctness by construction (more details on this paper in Section 7.1).

7.5.2. Name mobility

The article [44] studies the behavioural theory of $\pi\mathtt{P}$, a π-calculus featuring restriction as the only binder. In contrast with calculi such as Fusions and Chi, reduction in $\pi\mathtt{P}$ generates a preorder on names rather than an equivalence relation. Two characterisations of barbed congruence in $\pi\mathtt{P}$ are analyzed: the first is based on a compositional LTS, and the second is an axiomatisation. The results in this paper bring out basic properties of $\pi\mathtt{P}$, mostly related to the interplay between the restriction operator and the preorder on names.

7.5.3. Coinductive techniques

Coinductive techniques, notably those based on bisimulation, are widely used in concurrency theory to reason about systems of processes. The bisimulation proof method can be enhanced by employing “bisimulations up-to” techniques. A comprehensive theory of such enhancements has been developed for first-order (i.e., CCS-like) LTSs and bisimilarity, based on the notion of compatible function for fixed-point theory.

A proof method different from bisimulation is investigated in [46], [23]. This method is based on unique solution of special forms of inequations called contractions, and inspired by Milner’s theorem on unique solution of equations. The method is as powerful as the bisimulation proof method and its “up-to context” enhancements. The definition of contraction can be transferred onto other behavioural equivalences, possibly contextual and non-coinductive. This enables a coinductive reasoning style on such equivalences, either by applying the method based on unique solution of contractions, or by injecting appropriate contraction preorders into the bisimulation game. The technique can be applied both to first-order languages and to higher-order languages.

7.5.4. Expressiveness and decidability in actor-like systems

In [48], the limit of classical Petri nets is studied by discussing when it is necessary to move to the so-called Transfer nets, in which transitions can also move to a target place all the tokens currently present in a source place. More precisely, we consider a simple calculus of processes that interact by generating/consuming messages into/from a shared repository. For this calculus classical Petri nets can faithfully model the process behavior. Then we present a simple extension with a primitive allowing processes to atomically rename all the data of a given kind. We show that with the addition of such primitive it is necessary to move to Transfer nets to obtain a faithful modeling.

7.6. Constraint Programming

Participants: Roberto Amadini, Maurizio Gabbrielli, Jacopo Mauro.

The Constraint Programming (CP) paradigm is a general and powerful framework that enables to express relations between different entities in form of constraints that must be satisfied. The concept of constraint is ubiquitous and not confined to the sciences: constraints appear in every aspect of daily life in the form of requirements, obligations, or prohibitions. Historically, the FOCUS group has always had an interest in CP, see e.g., [53], [54]. The possible applications of CP are in fact numerous and disparate. As an example, CP can be used for the deployment of services in the cloud [21], [39].

CP essentially consists of two layers: (i) a modeling level, in which a real-life problem is identified, examined, and formalized into a mathematical model by human experts; (ii) a solving level, aimed at resolving as efficiently and comprehensively as possible the model defined in (i) by means of software agents called constraint solvers. Over the last years we dealt with a particular aspect of CP, that is, the so called portfolio approaches [12], [27], [10]. In a nutshell, a portfolio approach in CP can be seen as the problem of predicting which is (are) the best constraint solver(s) —among a portfolio of available solvers— for solving a given CP problem. A constraint solver that relies on a portfolio of underlying, individual solvers is also dubbed a portfolio solver.

Our studies on portfolio approaches lead to development of the SUNNY-CP portfolio solver [26], [25]. SUNNY-CP relies on underlying state-of-the-art constraint solvers for solving a given CP problem encoded in the MiniZinc language, nowadays a de-facto standard for modeling CP problems. Initially developed as a sequential solver [26], SUNNY-CP has been later on enhanced by enabling the simultaneous execution of its solvers on different cores [25]. This extension allowed SUNNY-CP to win the gold medal in the open track of 2015 MiniZinc Challenge [cite], the annual competition for CP solvers.

However, we did not restrict the work on portfolio approaches to the CP field only. Indeed, we also performed some preliminary studies for evaluating SUNNY (i.e., the algorithm on which SUNNY-CP relies) in other application domains like, e.g., Boolean satisfiability (SAT), Quantified Boolean Formula (QBF), and Answer-Set Programming (ASP) [47], [24].

8. Partnerships and Cooperations

8.1. National Initiatives

- REVER (Programming Reversible Recoverable Systems) is an ANR project that started on 1st December 2011 and with a 4-year duration. REVER aims to study the possibility of defining semantically well-founded and composable abstractions for dependable computing on the basis of a reversible programming language substrate, where reversibility means the ability to undo any distributed program execution, possibly step by step. The critical assumption behind REVER is that by adopting a reversible model of computation, and by combining it with appropriate notions of compensation and modularity, one can develop systematic and composable abstractions for recoverable and dependable systems. Main persons involved: Giachino, Lienhardt, Lanese, Laneve, Zavattaro.
- PACE (Processus non-standard: Analyse, Coinduction, et Expressivité) is an ANR project that started in 2013. The project targets three fundamental ingredients in theories of concurrent processes, namely coinduction, expressiveness, and analysis techniques. The project aims at processes that are beyond the realm of "traditional" processes. Specifically, the models studied exhibit one or more of the following features: probabilities, higher-order, quantum, constraints, knowledge, and

confidentiality. These models are becoming increasingly more important for today's applications. Coinduction is intended to play a pivotal role. Indeed, the approaches to expressiveness and the analysis techniques considered in the project are based on coinductive equalities. Main persons involved: Hirschkoff (project coordinator), Dal Lago, Lanese, Sangiorgi, Zavattaro.

- ELICA (Expanding Logical Ideas for Complexity Analysis) is an ANR project which started on October 2014 and that we will finish on September 2018. ELICA is a project about methodologies for the static analysis of programs as for their resource consumption. The project's aim is to further improve on logical methodologies for complexity analysis (type systems, rewriting, etc.). More specifically, one would like to have more powerful techniques with less false negatives, being able at the same time to deal with nonstandard programming paradigms (concurrent, probabilistic, etc.). Main persons involved: Avanzini, Cappai, Dal Lago, Hirschkoff, Martini, Sangiorgi.

8.2. European Initiatives

8.2.1. FP7 & H2020 Projects

- ENVISAGE (Engineering Virtualized Services) is a EU FP7 project, with starting date October 1st, 2013, and with a 3-year duration. The project is about model-based development of virtualized services, including tool support for resource analysis. Most Focus members are involved.

8.2.2. Collaborations in European Programs, except FP7 & H2020

- The ICT COST Action BETTY (Behavioural Types for Reliable Large-Scale Software Systems). initiated in October 2012 and with a 4-year duration, uses behavioural type theory as the basis for new foundations, programming languages, and software development methods for communication-intensive distributed systems. Behavioural type theory encompasses concepts such as interfaces, communication protocols, contracts, and choreographies. Main persons involved: Bravetti, Giachino, Hirschkoff, Lanese, Laneve, Mauro, Sangiorgi, Zavattaro.
- ICT COST Action IC1405 (Reversible computation - extending horizons of computing). Initiated at the end of April 2015 and with a 4-year duration, this COST Action studies reversible computation and its potential applications, which include circuits, low-power computing, simulation, biological modeling, reliability and debugging. Reversible computation is an emerging paradigm that extends the standard forwards-only mode of computation with the ability to execute in reverse, so that computation can run backwards as naturally as it can go forwards.

  Main persons involved: Giachino, Lanese (vice-chair of the action), Bernadet
- ICT COST Action IC1402 ARVI (Runtime Verification beyond Monitoring) Initiated in December 2014 and with a 4-year duration, this COST Action studies runtime verification, a computing analysis paradigm based on observing a system at runtime to check its expected behavior.

  Main persons involved: Lanese

8.2.3. Collaborations with Major European Organizations

Simone Martini is a member of the Executive Board of EQANIE (European Quality Assurance Network for Informatics Education), from October 2014.

We list here the cooperations and contacts with other groups, without repeating those already listed in previous sections.

- ENS Lyon (on concurrency models and resource control). Contact person(s) in Focus: Dal Lago, Martini, Sangiorgi, Vignudelli. Some visit exchanges during the year, in both directions. One joint PhD supervision (J.-M. Madiot).
- Inria EPI Spades (on models and languages for components, reversibility). Contact person(s) in Focus: Lanese. Some visit exchanges during the year, in both directions.

- Laboratoire d'Informatique, Université Paris Nord, Villetaneuse (on implicit computational complexity). Contact person(s) in Focus: Dal Lago, Martini. An Italian PhD student (Marco Solieri) is working on his PhD thesis with joint supervision (Martini, Guerrini).
- Institut de Mathématiques de Luminy, Marseille (on lambda-calculi, linear logic and semantics). Contact person(s) in Focus: Dal Lago, Martini.
- Team PPS, University of Paris-Diderot Paris 7 (on logics for processes, resource control). Contact person(s) in Focus: Dal Lago, Martini, Sangiorgi. Some short visits in both directions during the year.
- IRILL Lab, Paris (on models for the representation of dependencies in distributed package based software distributions). Contact person(s) in Focus: Mauro, Zavattaro. Some short visits in both directions during the year.
- EPI Carte, Inria-Nancy Grand Est and LORIA (on implicit computational complexity). Contact person(s) in Focus: Dal Lago.
- LMU Munich (M. Hofmann) (on Implicit computational complexity and IntML). Contact person(s) in Focus: Dal Lago.
- IMDEA Software, Madrid (G. Barthe) (on Implicit computational complexity for cryptography). Contact person(s) in Focus: Dal Lago,Sangiorgi. Some visits during 2015.
- Facultad de Informatica, Universidad Complutense de Madrid (on web services). Contact person(s) in Focus: Bravetti. Bravetti is an external collaborator in the project "ESTuDIo: ESpecificacion y Testing de sistemas altamente DIstribuidos" (Specification and Testing of Highly Distributed Systems) January 1, 2013 - December 31, 2015 (3 years), funded by the Spanish Ministerio de Economia y Competitividad.

8.3. International Initiatives

8.3.1. Inria Associate Teams not involved in an Inria International Labs

8.3.1.1. CRECOGI

Title: Concurrent, Resourceful and Effectful Computation, by Geometry of Interaction

International Partner (Institution - Laboratory - Researcher):

Todai (Japan) - Graduate School of Information Science and Technology - Ichiro HASUO

Start year: 2015

See also: http://crecogi.cs.unibo.it

Game semantics and geometry of interaction (GoI) are two closely related frameworks whose strength is to have the characters of both a denotational and an operational semantics. They offer a high-level, mathematical (denotational) interpretation, but are interactive in nature. The formalization in terms of movements of tokens through which programs communicate with each other can actually be seen as a low-level program. The current limit of GoI is that the vast majority of the literature and of the software tools designed around it have a pure, sequential functional language as their source language. This project aims at investigating the application of GoI to concurrent, resourceful, and effectful computation, thus paving a way to the deployment of GoI-based correct-by-construction compilers in real-world software developments in fields like (massively parallel) high-performance computing, embedded and cyberphysical systems, and big data. The presence of both the japanese GoI community (whose skills are centered around effects and coalgebras) and the french GoI community (more focused on linear logic and complexity analysis) will bring essential, complementary, ingredients.

8.3.2. Participation In other International Programs

Complexity Analysis of Higher-Order Rewrite Systems is an FWF (Austrian Science Fund, see http://www.fwf.ac.at/) project which is conducted in Bologna from April 2014 to April 2016. The project aim is the development of logical methodologies for the static resource analysis of higher-order rewrite systems, a formal model of computation that closely captures the evaluation semantics of functional programs. Particular attention is paid to automation, so that the developed complexity-techniques can be integrated into the Tyrolean Complexity Tool, a highly modular complexity analyser for rewrite systems.

Main persons involved: Avanzini, Dal Lago.

8.4. International Research Visitors

8.4.1. Visits of International Scientists

The following researchers have visited Focus for short periods; we list them together with the title of the talk they have given during their stay, or the topic discussed during their stay.

- Giovanni Pardini: "Lazy Security Controllers."
- Antonio Ravara: "Towards static deadlock resolution in the pi-calculus."
- Silvia Crafa: "Concurrency theory and concurrent languages: an evolutionary view of programming abstractions."
- Henning Kerstan: "Probabilistic Transition Systems."
- Jean-Bernard Stefani: "Location graphs: yet another global computing model (but with reason)."
- Isabel Oitavem: "P, NP and Pspace from a recursion-theoretic perspective."
- Reinhard Kahle: "Theories for Classes of Computational Complexity."
- Violet Ka I Pun: "Meeting Deadlines, Elastically."
- Volker Stolz: "Search-based composed refactorings."
- Giuseppe De Ruvo: "An Unusual Context for Formal Verification: Wiki Evolution."
- Manuel Mazzara: "Applications of reversibility".
- Claudio Antares Mezzina and Doriana Medic: "Calculi for reversibility".
- Thomas Leventis: " Theories in Probabilistic lambda-calculus."
- Mirko Viroli: "Perspectives on Aggregate Computing".
- Benoit Valiron and Claudia Faggian: "Resource control and linear logics".
- Beniamino Accattoli: "Abstract machines and resource control".
- Georg Moser, Thomas Powell and Stephane Gimenez: "Tools for compuational complexity”.
- Ichiro Hasuo, Akira Yoshimizu, and Ryo Tanaka: "Game semantics".
- Ales Bizjak: "Step-Indexed Logical Relations for Probability".
- Roberto Di Cosmo: "Preserving Software: challenges and opportunities for reproducibility of Science and Technology".

8.4.2. Visits to International Teams

January 14-28: Ugo Dal Lago visited Shanghai Jiaotong University, and collaborated with Yuxin Deng, as for the PACE project.

8.4.2.1. Sabbatical programme

Maurizio Gabbrielli is, since 15 September 2014, Head of the EIT ICT Labs Doctoral School with Paris as his principal location.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific events organisation

9.1.1.1. Chair of conference program committees

I. Lanese: Co-chair of 8th Interaction and Concurrency Experience (ICE 2015)

9.1.1.2. Member of the conference program committees

M. Bravetti: Int. Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2015); 9th IEEE Int. Conference on Big Data Science and Engineering (IEEE BigDataSE 2015); 4th European Conference on Service-Oriented and Cloud Computing (ESOCC 2015); 20th Int. Workshop on Formal Methods for Industrial Critical Systems (FMICS 2015); 6th Int. Conference on Fundamentals of Software Engineering (FSEN 2015); 13th Int. Conference on Software Engineering and Formal Methods (SEFM 2015).

U. Dal Lago: 18th Int. Conference on Foundations of Software Science and Computation Structures (FOSSACS 2015); 6th Workshop on Developments in Implicit Computational Complexity (DICE 2015); 31st Workshop on Mathematical Foundations of Programming Semantics (MFPS 2015); 42nd Int. Colloquium on Automata, Languages, and Programming (ICALP 2015); 11th Int. Workshop on Developments in Computational Models (DCM 2015); 16th Italian Conference on Theoretical Computer Science (ICTCS 2015); 24th EACSL Annual Conference on Computer Science Logic (CSL 2015).

D. Hirschkoff: 18th Int. Conference on Foundations of Software Science and Computation Structures (FoSSaCS), 2015.

I. Lanese: 12th Int. Conference on Formal Aspects of Component Software (FACS 2015); 7th Conference on Reversible Computation (RC 2015); 35th IFIP Int. Conference on Formal Techniques for Distributed Objects, Components and Systems (FORTE 2015); 11th Int. Workshop on Developments in Computational Models (DCM 2015); Special Session on Formal Approaches to Parallel and Distributed Systems (4PAD) of the 23rd Euromicro Int. Conference on Parallel, Distributed and Network-based Processing (PDP 2015).

S. Martini: 30th ACM/IEEE Symposium on Logic in Computer Science (LICS) 2015.

F. Montesi: 6th Int. Conference on Ambient Systems, Networks and Technologies (ANT 2015); 31st Annual ACM/SIGAPP Symposium On Applied Computing (SAC 2015) - Operating Systems track; 8th Programming Language Approaches to Concurrency- and Communication-cEntric Software (PLACES 2015). 8th Interaction and Concurrency Experience (ICE 2015).

D. Sangiorgi: 20th Int. Symposium on Fundamentals of Computation Theory 2015 (FCT); 10th Int. Andrei Ershov Memorial Conference (PSI 2015); 10th Int. Conference on Language and Automata Theory and Applications.

V. Vignudelli: 8th Interaction and Concurrency Experience (ICE 2015).

G. Zavattaro: 17th Int. Conference on Formal Engineering Methods (ICFEM 2015); 12th Int. Colloquium on Theoretical Aspects of Computing (ICTAC 2015); 8th IEEE Int. Conference on Service Oriented Computing and Applications (SOCA 2015); 4th European Conference on Service-oriented and Cloud Computing (ESOCC'15); 5th Int. Workshop on Adaptive Services for the Future Internet (WAS4FI 2015); 3rd IEEE Int. Workshop on Formal Methods Integration (FMi 2015); 20th Int. Symposium on Formal Methods (FM2015); Int. Conference on Cloud Computing Technologies and Applications (CLOUDTECH 2015); 30th ACM/SIGAPP Symposium On Applied Computing - track on Software Verication and Testing (ACM-SAC 2015).

9.1.1.3. Member of the editorial boards

U. Dal Lago: Logical Methods in Computer Science.

M. Gabbrielli: Int. Journal Theory and Practice of Logic Programming.

C. Laneve: Frontiers in ICT (Section Formal Methods).

I. Lanese: Editor in chief of the Open Journal of Communications and Software (Scientific Online).

F. Montesi: Guest editor for "Special issue on Service-Oriented Architecture and Programming (SOAP 2013)". Science of Computer Programming.

D. Sangiorgi: Logical Methods in Computer Science, Acta Informatica, Distributed Computing, RAIRO Theoretical Informatics and Applications.

9.1.2. Invited talks

M. Bravetti: 14th Int. Workshop on Foundations of Coordination Languages and Self-Adaptive Systems (FOCLASA 2015).

U. Dal Lago: gave an invited course “From Implicit Complexity to Quantitative Resource Analysis” at the Warsaw’s Open PhD lectures in Computer Science.

C. Laneve: Int. Symposium on Web Services, Formal Methods and Behavioural Types (WS-FM/BEAT 2015).

S. Martini: Colloque “Computing, Logic and Types”, Université Jean Moulin Lyon 3.

D Sangiorgi: DIMAP Logic Day 2015, Centre for Discrete Mathematics and its Applications, University of Warwick; Joint 22th Int. Workshop on Expressiveness in Concurrency and 12th Workshop on Structural Operational Semantics, Madrid; 12th Int. Colloquium on Theoretical Aspects of Computing (ICTAC 2015), Cali, Colombia.

G. Zavattaro: 26th Conference on Concurrency Theory (CONCUR’15).

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

- Mario Bravetti

  Undergraduate: “Tecnologie Web”, 66 hours, 3rd year, University of Bologna, Italy.

  Master: “Linguaggi di Programmazione e Modelli Computazionali”, 120 hours, 2nd year, University of Bologna, Italy.

- Ugo Dal Lago

  Undergraduate: “Introduction to Programming in Python”, 20 hours, 1st year, University of Bologna, Italy.

  Undergraduate: “Optimization”, 36 hours, 2nd year, University of Bologna, Italy.

  Master: “Cryptography”, 36 Hours, 2nd Year, University of Bologna, Italy.

- Maurizio Gabbrielli

  Undergraduate: “Programming languages”, 40 hours, 2nd year, University of Bologna, Italy.

  Master: “Artificial Intelligence”, 60 hours, 2nd year, University of Bologna, Italy.

- Elena Giachino

  Undergraduate: “Programmazione”, 40 hours, 1st year, University of Bologna, Italy.

- Saverio Giallorenzo

  Undergraduate: “Laboratorio di Operating Systems”, 40 hours, 2nd year, University of Bologna, Italy.

- Ivan Lanese

  Undergraduate: “Programmazione”, 32 hours, 1st year, University of Bologna, Italy.

  Master: “Ingegneria del Software Orientata ai Servizi”, 22 hours, 2nd year, University of Bologna, Italy.

- Cosimo Laneve

  Undergraduate: “Programmazione”, 70 hours, 1st year, University of Bologna, Italy.

  Master: “Analisi di Programmi”, 42 hours, 1st year, University of Bologna, Italy.

- Simone Martini

  Undergraduate: “Introduction to programming in Python”, 58 hours, 1st year, University of Bologna, Italy.

Undergraduate: “Computer abilities for biologists”, 8 hours, 1st year, University of Bologna, Italy.

Master: “Logical Foundations of Computer Science”, 48 hours, 2nd year, University of Bologna, Italy.

Undergraduate: “Programming languages”, 50 hours, 2nd year, University of Bologna, Italy.

- Jacopo Mauro

  Undergraduate: “Algoritmi e strutture dati”, 20 hours, 1st year, University of Bologna, Italy.

- Giulio Pellitta

  Undergraduate: “Tutorato informatico trasversale”, 100 hours, Degree in Quantitative Finance, University of Bologna, Italy.

  Undergraduate: “Attivita’ di supporto alla didattica universitaria di Informatica nell’ambito della Piattaforma di e-learning A3”, 120 hours, 1st year, University of Bologna, Italy.

  Undergraduate: “Idoneita’ Informatica”, 50 hours, 1st year, University of Bologna, Italy.

  Undergraduate: “C.I. Fisica, Informatica e Statistica - Modulo 2 Informatica”, 12 hours, 1st year, University of Bologna, Italy.

- Davide Sangiorgi

  Undergraduate: “Operating Systems”, 110 hours, 2nd year, University of Bologna, Italy.

  Master: “Models for concurrency”, 15 hours, 2nd year, University of Bologna, Italy.

- Marco Solieri

  Undergraduate: “Programmation impérative”, 33 hours, 1st year, Inst. Galilée, Université Paris 13.

  Undergraduate: “Structures de données”, 12 hours, 2nd year, Cursus preparatoire integré ingénieurs, Sup Galilée, Université Paris 13.

  Undergraduate: “Programmation fonctionnelle”, 14 hours, 2nd year, Inst. Galilée, Université Paris 13

  Undergraduate: “Programmation web”, 20 hours, 3rd year, Inst. Galilée, Université Paris 13

  Undergraduate: “Administration système” 30 hours, 3rd year, Inst. Galilée, Université Paris 13

  Master: “Internet, Intranet, Web”, 35 hours, 2nd year, Ecole d’ingénieurs Sup Galilée, Université Paris 13

  Master: “Administration systeme Unix”, 44 hours, 3rd year, Ecole d’ingénieurs Sup Galilée, Universitše Paris 13

- Gianluigi Zavattaro

  Master: “Linguaggi di Programmazione e Modelli Computazionali”, 120 hours, 1st year, University of Bologna, Italy.

9.2.2. Supervision

PhD thesis completed in 2015:

- Jean-Marie Madiot, “Higher-order languages: dualities and bisimulation enhancements”, jointly supervised by D. Sangiorgi and D. Hirschkoff. March 2015 [11].
- R. Amadini. “Portfolio Approaches in Constraint Programming”, University of Bologna, June 2015 [10].

Below are the details on the PhD students in Focus: starting date, topic or provisional title of the thesis, supervisor(s). These are all PhDs in progress.

- Ferdinanda Camporesi, January 2009, “Analysis of system biology”, R. Cousot (Ecole Normale, Paris) and M. Gabbrielli
- Alberto Cappai, January 2013; “Security, ICC and probabilistic bisimulation”, U. Dal Lago.
- Abel Garcia, January 2014, “Analysis of Cloud Computing Systems”, C. Laneve.
- Francesco Gavazzo, October 2015, “Causality in coinduction”, D. Sangiorgi.
- Saverio Giallorenzo, January 2013, “Real-World Choreographies”, M. Gabbrielli and F. Montesi
- Vincenzo Mastandrea, October 2015, “Deadlock analysis in ASP”, Cosimo Laneve (with Ludovic Henrio, CNRS Sophia Antipolis).
- Alessandro Rioli, January 2012; Bisimulation equivalence in quantum lambda calculi, U. Dal Lago.
- Marco Solieri, December 2011; “Géometrie de l’interaction, réseaux, et lambda-calcul”, S. Guerrini (Paris Nord) and S. Martini.
- Valeria Vignudelli, January 2014, “Probabilities in Higher-Order languages”, D. Sangiorgi.

9.2.3. Juries

Dal Lago has been member of the PhD evaluation committee for Erika De Benedetti (January 2015, University of Torino).

Laneve has been member of the PhD evaluation committee for Abiar Al-Homaimeedi, King’s College London, June 2015

Simone Martini has been rapporteur and member of the PhD evaluation committee for Matthieu Perrinel (Ecole Normale Supèrieure de Lyon); member of the PhD evaluation committee for Paolo Pistone (Università Roma Tre) and Federica Panarotto (Università di Verona).

9.3. Popularization

Simone Martini has carried out extended work of scientific divulgation:

- member of the technical committee of Olimpiadi del Problem Solving (at Italian Ministry of Education), http://www.olimpiadiproblemsolving.com;
- member of the technical committee of two national projects of Italian Ministry of Education, founded by ESF - European Social Fund: ‘Robotica educativa’, and ‘Informatica nei licei’;
- various talks at institutes and workshops on the teaching methods for Computer Science.
- coordinator of some initiatives for the ‘Hour of Code’, see https://italia.code.org/ and http://www.programmailfuturo.it.

D. Hirschkoff takes part in several popularization activities in schools, in Lyon (association "Maths en Jeans").

9.3.1. Other duties

Simone Martini is a member of the Board of CINI (Italian National Interuniversity Consortium for Informatics), designated by the Ministry for Semplificazione e Pubblica Amministrazione, from 2015.

Simone Martini has been elected Head of the Department of Computer Science and Engineering, Università di Bologna, for the term 2015-2018.

Davide Sangiorgi is coordinator of undergraduate studies at the Department of Computer Science and Engineering, Università di Bologna (Informatica per il Management).

10. Bibliography

Major publications by the team in recent years

[1] M. Bravetti, G. Zavattaro. *A Foundational Theory of Contracts for Multi-party Service Composition*, in "Fundam. Inform.", 2008, vol. 89, n^{o} 4, pp. 451-478

[2] N. Busi, M. Gabbrielli, G. Zavattaro. *On the expressive power of recursion, replication and iteration in process calculi*, in "Mathematical Structures in Computer Science", 2009, vol. 19, n^{o} 6, pp. 1191-1222

[3] P. Coppola, S. Martini. *Optimizing optimal reduction: A type inference algorithm for elementary affine logic*, in "ACM Trans. Comput. Log.", 2006, vol. 7, n^{o} 2, pp. 219-260

[4] M. Gabbrielli, S. Martini. *Programming Languages: Principles and Paradigms*, Springer, 2010

[5] D. Hirschkoff, É. Lozes, D. Sangiorgi. *On the Expressiveness of the Ambient Logic*, in "Logical Methods in Computer Science", 2006, vol. 2, n^{o} 2

[6] U. D. Lago, M. Gaboardi. *Linear Dependent Types and Relative Completeness*, in "Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science, LICS 2011", IEEE Computer Society, 2011, pp. 133-142, https://hal.archives-ouvertes.fr/hal-00906347

[7] I. Lanese, C. A. Mezzina, A. Schmitt, J.-B. Stefani. *Controlling Reversibility in Higher-Order Pi*, in "Proc. of CONCUR 2011", J.-P. Katoen, B. König (editors), Lecture Notes in Computer Science, Springer, 2011, vol. 6901, pp. 297-311, http://www.cs.unibo.it/~lanese/publications/fulltext/concur2011.pdf.gz

[8] F. Montesi, C. Guidi, G. Zavattaro. *Composing Services with JOLIE*, in "Fifth IEEE European Conference on Web Services (ECOWS 2007)", 2007, pp. 13-22

[9] D. Sangiorgi. *An introduction to Bisimulation and Coinduction*, Cambridge University Press, 2012

Publications of the year

Doctoral Dissertations and Habilitation Theses

[10] R. Amadini. *Portfolio Approaches in Constraint Programming*, University of Bologna, June 2015, https://hal.inria.fr/tel-01227582

[11] J.-M. Madiot. *Higher-order languages : dualities and bisimulation enhancements*, Ecole normale supérieure de lyon - ENS LYON, March 2015, https://tel.archives-ouvertes.fr/tel-01141067

Articles in International Peer-Reviewed Journals

[12] R. Amadini, M. Gabbrielli, J. Mauro. *Portfolio Approaches for Constraint Optimization Problems*, in "Annals of Mathematics and Artificial Intelligence", 2015 [*DOI :* 10.1007/s10472-015-9459-5], https://hal.inria.fr/hal-01227939

[13] U. DAL LAGO, P. PARISEN. *A Higher-Order Characterization of Probabilistic Polynomial Time*, in "Information and Computation", 2015, vol. 241, pp. 114-141 [*DOI :* 10.1016/J.IC.2014.10.009], https://hal.inria.fr/hal-01231752

[14] M. DALLA PREDA, M. GABBRIELLI, S. GIALLORENZO, I. LANESE, J. MAURO. *Developing correct, distributed, adaptive software*, in "Science of Computer Programming", 2015 [*DOI :* 10.1016/J.SCICO.2013.11.019], https://hal.inria.fr/hal-01227610

[15] E. GIACHINO, C. LANEVE, M. LIENHARDT. *A framework for deadlock detection in core ABS*, in "Software and Systems Modeling", 2015 [*DOI :* 10.1007/S10270-014-0444-Y], https://hal.inria.fr/hal-01229046

[16] C. LANEVE, L. PADOVANI. *An Algebraic Theory for Web Service Contracts*, in "Formal Aspects of Computing", July 2015, vol. 27, n^{o} 4, 27 p. [*DOI :* 10.1007/S00165-015-0334-2], https://hal.inria.fr/hal-01229401

[17] T. A. LASCU, J. MAURO, G. ZAVATTARO. *Automatic deployment of component-based applications*, in "Science of Computer Programming", 2015 [*DOI :* 10.1016/J.SCICO.2015.07.006], https://hal.inria.fr/hal-01233485

[18] M. SOLIERI. *Geometry of Resource Interaction - A Minimalist Approach*, in "Electronic Proceedings in Theoretical Computer Science", February 2015, vol. 176, pp. 79-94 [*DOI :* 10.4204/EPTCS.176.7], https://hal-univ-paris13.archives-ouvertes.fr/hal-01091754

Invited Conferences

[19] M. BRAVETTI. *Towards Dynamic Updates in Service Composition*, in "FOCLASA 2015 - Proceedings of 14th International Workshop on Foundations of Coordination Languages and Self-Adaptive Systems", Madrid, Spain, EPTCS - Electronic Proceedings in Theoretical Computer Science, September 2015, https://hal.inria.fr/hal-01231465

[20] R. CRUBILLÉ, U. DAL LAGO, D. SANGIORGI, V. VIGNUDELLI. *On Applicative Similarity, Sequentiality, and Full Abstraction*, in "Correct System Design. Symposium in Honor of Ernst-Rüdiger Olderog on the Occasion of His 60th Birthday", Oldenburg, Germany, September 2015 [*DOI :* 10.1007/978-3-319-23506-6_7], https://hal.inria.fr/hal-01229398

[21] R. DI COSMO, M. LIENHARDT, J. MAURO, S. ZACCHIROLI, G. ZAVATTARO, J. ZWOLAKOWSKI. *Automatic Application Deployment in the Cloud: from Practice to Theory and Back*, in "CONCUR", Madrid, Spain, September 2015, https://hal.inria.fr/hal-01233478

[22] I. LANESE, F. MONTESI, G. ZAVATTARO. *The Evolution of Jolie*, in "Essays Dedicated to Martin Wirsing on the Occasion of His Retirement from the Chair of Programming and Software Engineering", Munich, Germany, Lecture Notes in Computer Science, March 2015, n^{o} 8950, pp. 506-521 [*DOI :* 10.1007/978-3-319-15545-6_29], https://hal.inria.fr/hal-01227623

[23] D. SANGIORGI. *The Proof Technique of Unique Solutions of Contractions*, in "12th International Colloquium on Theoretical Aspects of Computing", Cali, Colombia, SPRINGER (editor), October 2015, vol. Lecture Notes in Computer Science, n^{o} 9399, pp. 63–68 [*DOI :* 10.1007/978-3-319-25150-9_5], https://hal.inria.fr/hal-01227569

International Conferences with Proceedings

[24] R. Amadini, F. Biselli, M. Gabbrielli, T. Liu, J. Mauro. *Feature Selection for SUNNY: a Study on the Algorithm Selection Library*, in "ICTAI", Vietri sul Mare, Italy, November 2015, https://hal.inria.fr/hal-01227600

[25] R. Amadini, M. Gabbrielli, J. Mauro. *A Multicore Tool for Constraint Solving*, in "IJCAI 2015 - Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence", Buenos Aires, Argentina, July 2015, pp. 232-238, https://hal.inria.fr/hal-01227592

[26] R. Amadini, M. Gabbrielli, J. Mauro. *SUNNY-CP : a Sequential CP Portfolio Solver*, in "SAC", Salamanca, Spain, April 2015 [*DOI :* 10.1145/2695664.2695741], https://hal.inria.fr/hal-01227589

[27] R. Amadini, M. Gabbrielli, J. Mauro. *Why CP Portfolio Solvers Are (under)Utilized? Issues and Challenges*, in "LOPSTR", Siena, Italy, July 2015, https://hal.inria.fr/hal-01227598

[28] M. Avanzini, U. Dal Lago. *On Sharing, Memoization, and Polynomial Time*, in "Proceedings of STACS 2015", Munich, Germany, 2015, vol. 62 [*DOI :* 10.4230/LIPIcs.STACS.2015.62], https://hal.inria.fr/hal-01231816

[29] M. Avanzini, U. Dal Lago, G. Moser. *Analysing the Complexity of Functional Programs: Higher-Order Meets First-Order*, in "ICFP 2015", Vancouver, Canada, August 2015 [*DOI :* 10.1145/2784731.2784753], https://hal.inria.fr/hal-01231809

[30] M. Avanzini, C. Sternagel, R. Thiemann. *Certification of Complexity Proofs using CeTA*, in "26th International Conference on Rewriting Techniques and Applications", Warsaw, Poland, June 2015 [*DOI :* 10.4230/LIPIcs.RTA.2015.23], https://hal.inria.fr/hal-01252000

[31] F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, U. De 'liguoro. *Retractable Contracts*, in "Eighth International Workshop on Programming Language Approaches to Concurrency- and Communication-cEntric Software", London, United Kingdom, Electronic Proceedings in Theoretical Computer Science, April 2015, https://hal.inria.fr/hal-01227693

[32] A. Cappai, U. Dal Lago. *On Equivalences, Metrics, and Polynomial Time*, in "FCT 2015", Gdansk, Poland, August 2015, vol. 9210, pp. 311-323 [*DOI :* 10.1007/978-3-319-22177-9_24], https://hal.inria.fr/hal-01231790

[33] R. Crubillé, U. Dal Lago. *Metric Reasoning About λ-Terms: The Affine Case*, in "LICS 2015", Kyoto, Japan, July 2015 [*DOI :* 10.1109/LICS.2015.64], https://hal.inria.fr/hal-01231814

[34] U. Dal Lago, C. Faggian, B. Valiron, A. Yoshimizu. *Parallelism and Synchronization in an Infinitary Context*, in "LICS 2015", Kyoto, Japan, July 2015 [*DOI :* 10.1109/LICS.2015.58], https://hal.inria.fr/hal-01231813

[35] U. Dal Lago, A. Rioli. *Applicative Bisimulation and Quantum λ-Calculi*, in "FSEN 2015", Tehran, Iran, April 2015, vol. 9392, pp. 54-68 [*DOI :* 10.1007/978-3-319-24644-4_4], https://hal.inria.fr/hal-01231800

[36] M. Dalla Preda, M. Gabbrielli, S. Giallorenzo, I. Lanese, J. Mauro. *Dynamic Choreographies Safe Runtime Updates of Distributed Applications*, in "COORDINATION 2015", Grenoble, France, June 2015 [*DOI :* 10.1007/978-3-319-19282-6_5], https://hal.inria.fr/hal-01227612

[37] S. De Gouw, M. Lienhardt, J. Mauro, B. Nobakht, G. Zavattaro. *On the Integration of Automatic Deployment into the ABS Modeling Language*, in "4th European Conference on Service-Oriented and Cloud computing, ESOCC 2015", Taormina, Italy, S. Dustdar, F. Leymann, M. Villari (editors), LNCS, Springer, September 2015, vol. 9306, pp. 49–64 [*DOI :* 10.1007/978-3-319-24072-5_4], https://hal.inria.fr/hal-01233474

[38] Y. Deng, Y. Feng, U. Dal Lago. *On Coinduction and Quantum Lambda Calculi*, in "Proceedings of CONCUR 2015", Madrid, Spain, 2015, vol. 42 [*DOI :* 10.4230/LIPIcs.CONCUR.2015.427], https://hal.inria.fr/hal-01231782

[39] R. Di Cosmo, A. Eiche, J. Mauro, S. Zacchiroli, G. Zavattaro, J. Zwolakowski. *Automatic Deployment of Services in the Cloud with Aeolus Blender*, in "ICSOC 2015: 13th International Conference on Service Oriented Computing", Goa, India, A. Barros, D. Grigori, N. C. Narendra, H. K. Dam (editors), Springer, November 2015, vol. 9435, pp. 397–411 [*DOI :* 10.1007/978-3-662-48616-0_28], https://hal.inria.fr/hal-01233489

[40] A. Garcia, C. Laneve, M. Lienhardt. *Static analysis of cloud elasticity*, in "17th International Symposium on Principles and Practice of Declarative Programming", Siena, Italy, Moreno Falaschi and Elvira Albert, July 2015, 12 p. [*DOI :* 10.1145/2790449.2790524], https://hal.inria.fr/hal-01229424

[41] E. Giachino, E. Broch Johnsen, C. Laneve, K. I. Pun. *Time complexity of concurrent programs*, in "FACS 2015", Niterói, Rio de Janeiro, Brazil, October 2015, https://hal.inria.fr/hal-01229068

[42] E. Giachino, I. Lanese, C. A. Mezzina, F. Tiezzi. *Causal-Consistent Reversibility in a Tuple-Based Language*, in "PDP 2015 - 23rd Euromicro International Conference on Parallel, Distributed and Network-Based Processing", Turku, Finland, IEEE, March 2015, pp. 467 - 475 [*DOI :* 10.1109/PDP.2015.98], https://hal.inria.fr/hal-01227615

[43] C. Guidi, S. Giallorenzo, M. Gabbrielli. *Towards a Composition-based APIaaS Layer*, in "CLOSER 2014", Barcelona, Spain, April 2015 [*DOI :* 10.1007/978-3-319-07593-8_17], https://hal.inria.fr/hal-01090125

[44] *Best Paper*
D. Hirschkoff, J.-M. Madiot, X. Xu. *A behavioural theory for a π-calculus with preorders*, in "Proceedings of 6th IPM International Conference on Fundamentals of Software Engineering (FSEN 2015)", Teheran, Iran, LNCS, Springer Verlag, April 2015, This paper obtained the best paper award at the conference FSEN'2015. A revised version of this paper has been published in JLAMP (Journal of Logical and Algebraic Methods in Programming), Volume 84, Issue 6, November 2015, Pages 806–825. 2014, https://hal.archives-ouvertes.fr/hal-01246094.

[45] J. Mauro, G. Zavattaro. *On the Complexity of Reconfiguration in Systems with Legacy Components*, in "40th International Symposium on Mathematical Foundations of Computer Science, MFCS 2015", Milan, Italy, G. F. Italiano, G. Pighizzini, D. Sannella (editors), LNCS, Springer, August 2015, vol. 9234, pp. 382–393 [*DOI :* 10.1007/978-3-662-48057-1_30], https://hal.inria.fr/hal-01233482

[46] D. Sangiorgi. *Equations, contractions, and unique solutions*, in "POPL 2015 - Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Language", Mumbai, India, ACM, January 2015 [*DOI :* 10.1145/2676726.2676965], https://hal.archives-ouvertes.fr/hal-01089205

National Conferences with Proceedings

[47] R. Amadini, F. Biselli, M. Gabbrielli, T. Liu, J. Mauro. *SUNNY for Algorithm Selection: A Preliminary Study*, in "CILC", Genova, Italy, July 2015, https://hal.inria.fr/hal-01227595

Scientific Books (or Scientific Book chapters)

[48] G. Zavattaro. *When to Move to Transfer Nets On the limits of Petri nets as models for process calculi*, in "Programming Languages with Applications to Biology and Security", Lecture Notes in Computer Science, Springer, 2015, vol. 9465, pp. 339-353 [*DOI :* 10.1007/978-3-319-25527-9_22], https://hal.inria.fr/hal-01233419

Research Reports

[49] P. Baillot, G. Barthe, U. Dal Lago. *Implicit Computational Complexity of Subrecursive Definitions and Applications to Cryptographic Proofs (Long version)*, ENS Lyon, September 2015, https://hal.archives-ouvertes.fr/hal-01197456

[50] S. De Gouw, M. Lienhardt, J. Mauro, B. Nobakht, G. Zavattaro. *On the Integration of Automatic Deployment into the ABS Modeling Language*, Inria Sophia Antipolis, July 2015, https://hal.inria.fr/hal-01170926

[51] R. Di Cosmo, A. Eiche, J. Mauro, G. Zavattaro, S. Zacchiroli, J. Zwolakowski. *Automatic Deployment of Software Components in the Cloud with the Aeolus Blender*, Inria Sophia Antipolis, January 2015, https://hal.inria.fr/hal-01103806

[52] J. Mauro, G. Zavattaro. *On the Complexity of Reconfiguration in Systems with Legacy Components*, Inria Sophia Antipolis, June 2015, https://hal.inria.fr/hal-01160369

References in notes

[53] F. De Boer, M. M. Jaghoori, C. Laneve, G. Zavattaro. *Decidability Problems for Actor Systems*, in "Logical Methods in Computer Science", 2014, vol. 104, pp. 1 - 29 [*DOI :* 10.2168/LMCS-10(4:5)2014], https://hal.inria.fr/hal-01090952

[54] C. D. Giusto, M. Gabbrielli, M. C. Meo. *On the Expressive Power of Multiple Heads in CHR*, in "ACM Transactions on Computational Logic", 2012, vol. 13, n^{o} 1 [*DOI :* 10.1145/2071368.2071374], https://hal.inria.fr/hal-00909390