INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team focus

Foundations of Component-based Ubiquitous Systems

Sophia Antipolis - Méditerranée

Theme : Distributed Systems and Services

Activity Report

2010

Table of contents

1. Team

Faculty Members

Mario Bravetti ["Ricercatore", Bologna]
Ugo Dal Lago ["Ricercatore", Bologna]
Maurizio Gabbrielli ["Prof. Ordinario", Bologna]
Cosimo Laneve ["Prof. Ordinario", Bologna]
Simone Martini ["Prof. Ordinario", Bologna]
Davide Sangiorgi [Team leader, "Prof. Ordinario", Bologna, HdR]
Gianluigi Zavattaro ["Prof. Associato", Bologna]

External Collaborators

Daniel Hirschkoff [ENS Lyon, HdR]
Marco Pistore [Senior Researcher, FBK Trento (I)]

PhD Students

Ferdinanda Camporesi [joint supervision with Paris 7]
Romain Demangeon [joint supervision with ENS Lyon, completed November 2010]
Jacopo Mauro [Bologna]
Fabrizio Montesi [joint supervision with IT University of Copenhagen]
Paolo Parisen Toldin [Bologna]
Jorge Perez Parra [Bologna, completed May 2010]

Post-Doctoral Fellows

Marco Gaboardi [Bologna]
Elena Giachino [EU project Hats]
Claudio Guidi [presently at University of Padova, Italy]
Ivan Lanese [Bologna]
Michaël Lienhardt [EU project Hats]
Mila Della Preda [Emilia Romagna region]

Administrative Assistant

Nathalie Woodward [shared with another team]

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 are 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. 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.

2.2. Highlights

- During the period 2006-2010, the Focus members have had 5 papers in the prestigious conference IEEE Symposium on Logic in Computer Science (LICS).
- Development and implementation of the programming language Jolie, for Service Oriented Computing.
- Launch of a start-up (Italiana Software) around Jolie.
- M. Gabbrielli and S. Martini published in 2010 a book on Programming Languages with Springer, in the Series Undergraduate Topics in Computer Science, UTiCS. The book is designed to cover the fundamental arguments in programming languages in a way accessible to undergraduate students in computer science. It is one of a few examples of a computer science book first published in Italian and then translated in English, on request of the publisher.

3. Scientific Foundations

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.

The sections below provide some more details on why process calculi, λ-calculi, and related techniques, should be useful for CBUS.

3.2. Foundational calculi and interaction

Modern distributed systems have witnessed a clear shift towards interaction and conversations as basic building blocks for software architects and programmers. The systems are made by components, that are supposed to interact and carry out dialogues in order to achieve some predefined goal; Web services are a good example of this. Process calculi are models that have been designed precisely with the goal of understanding interaction and composition. The theory and tools that have been developed on top of process calculi can set a basis from which attacking CBUS challenges. Indeed industrial proposals of languages for Web services such as BPEL are strongly inspired by process calculi, notably the π-calculus.

3.3. Type systems and logics

Type systems and logics for reasoning on computations are among the most successful outcomes in the history of the research in λ-calculus and (more recently) in process calculi. Type systems can also represent a powerful means of specifying dialogues among components of CBUS. For instance—again referring to Web services—current languages for specifying interactions only express basic connectivity, ignoring causality and timing aspects (e.g., an intended order on the messages), and the alternative is to use Turing Complete languages that are however undecidable. Types can come at hand here: they can express causality and order information on messages [59], [58][17], while remaining decidable systems.

3.4. Implicit computational complexity

A number of elegant and powerful results have been recently obtained in implicit computational complexity in the λ-calculus in which ideas from Linear Logics enable a fine-grained control over computations. This experience can be profitable when tackling issues of CBUS related to resource consumption, such as resources allocation, access to resources, certification of bounds on resource consumption (e.g., ensuring that a service will answer to a request in time polynomial with respect to the size of the input data).

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

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”. Strictly speaking Service Oriented Computing is constructed around the Web Service technology: it allows web architects to produce new services by assembling existing services available from the shelves (in web development, the term ‘mashup’ is sometimes employed for this). According to the specific task at hand, a search is performed for suitable services to provide the needed capabilities.

4.3. Software Product Lines

A Software Product Line is a set of software systems that together address a particular market segment or fulfill a particular mission. Today, Software Product Lines are successfully applied in a range of industries, including telephony, medical imaging, financial services, car electronics, and utility control [60]. Customization and integration are keywords in Software Product Lines: a specific system in the family is constructed by selecting its properties (often technically called “features”), and, following such selection, by customizing and integrating the needed components and deploying them on the required platform.

5. Software

5.1. Jolie

Members of Focus have recently developed Jolie [7] (Java Orchestration Language Interpreter Engine, see http://www.jolie-lang.org/). Jolie is a service programming language for which we have already implemented an interpreter and a run-time environment. Jolie can be used to program services that interact over the Internet using different communication protocols, including SOAP that is the standard communication protocol for Web Services. 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. A spin-off, called “Italiana Software”, has been launched around Jolie, its general aim is to transfer the expertise in formal methods for Web Services matured in the last few years onto Service Oriented Business Applications. The spin-off is a software producer and consulting company that offers service-oriented solutions (for instance, a “‘single sign-on” application) based on the Jolie language.

5.2. Pi-Duce

`PiDuce` (see http://www.cs.unibo.it/PiDuce/) is a prototype for experimenting Web services technologies by relying on solid theories about process calculi and formal languages for XML documents and schemas [3]. `PiDuce` consists of a programming language and a distributed runtime environment. The language features values and datatypes that extend XML documents and schemas with channels, an expressive type system with subtyping, a pattern matching mechanism for deconstructing XML values, and control constructs that are based on the asynchronous version of Milner’s π-calculus. The runtime environment supports the execution of PiDuce processes over networks by relying on state-of-the-art technologies, such as XML schema and WSDL, thus enabling interoperability with existing Web services.

5.3. IntML

IntML is a functional programming language guaranteeing sublinear space bounds for all programs [32]. Sublinear space bounds can only be achieved if input is accessed interactively, since the computation space used by the underlying program must be asymptotically smaller than the size of the input. It is natural, then, to structure computation as a bidirectional process. *IntML* is able to guarantee those bounds by a type discipline inspired by linear logic and by translating programs into message passing networks, for which space consumption corresponds to the size of exchanged messages. The obtained language is able to naturally express logspace algorithms like graph algorithms (e.g. the Cook-McKenzie test for acyclicity), arithmetical algorithms and sorting algorithms.

5.4. JoRBA

JoRBA [41] is a framework for programming dynamically adaptable applications in a service oriented scenario. The main aim is to make those applications adaptable to new environment conditions and new user desires that were unexpected when the application has been developed, deployed and even started. The basic idea of the approach is that an adaptable application provides a suitable interface for interacting with some adaptation managers in the environment. Those adaptation managers include rules to update specific activities in the code of the running application, so to allow it to satisfy the new requirements.

The JoRBA prototype shows the suitability of the approach, providing a skeleton for adaptable applications, the adaptation middleware and a skeleton for adaptation managers and rules. The JoRBA prototype has been implemented in the Jolie language, since Jolie primitives for service oriented communication are suitable for implementing the interactions between the adaptation middleware and the adaptable applications.

5.5. Java-based Comet Architectures for Web 2.0

Software architectures based on Java have been studied and realized in several Master theses to provide libraries and skeleton implementations for front-end (client-side) and back-end (server-side) of internet applications capable of managing asynchronous messaging from the back-end (thus supporting real-time cooperative applications). The libraries developed form the basis for the middleware of a proposed Web OS implementation [51].

6. New Results

6.1. Contracts and sessions

Participants: Maurizio Gabbrielli, Elena Giachino, Jacopo Mauro, Fabrizio Montesi, Marco Pistore, Gianluigi Zavattaro.

Contracts are descriptions of the functionalities offered by a component or a service, and of the way these functionalities may be accessed by clients. A contract may include a description of the component capabilities, place constraints on their usage, as well as declare preferences, entitlements and credentials. When a client wants to use one of the functionalities offered, it engages a dialogue (e.g., a sequence of interactions) with the servers; this is usually called a *session*.

The expected dialogue in a session can be specified by means of types, the *session types*. We have studied [38] the use of session types in object-oriented languages; the types that we have studied enable subtyping, session overloading, and type safety in the sense that the evaluation of well-typed terms cannot produce deadlock.

In service-oriented architectures, the mechanism that allows to manage sessions and, in particular, to assign incoming messages to the correct sessions, is critical for effciency and performance. A well-known solution to this problem, first introduced by BPEL, makes use of *correlation sets*. Intuitively these distinguish different sessions by means of the values for some specic variables which are present also in messages, thus allowing for their routing to sessions on the basis of these values. We have proposed [54] a new algorithm for managing correlation sets, based on radix trees, that is more efficient that existing ones. We are currently implementing the algorithm in the Jolie language.

Previous formal models for contracts consider unidirectional send and receive operations. In [23] we have studied primitives for contracts with bidirectional request-response operations, one that takes inspiration from the abstract service interface language WSCL and another one inspired by Abstract WS-BPEL [57]. We have studied properties of client-service compliance in the two models (e.g., decidability).

Contracts are also related to service discovery, since the information in a contract could be used to search for a needed service. The paper [40] studies service discovery, from a different angle: discovery is centered around the activities of the user and the information that he/she has available (e.g., a specific location or situation) rather than the goals that a given service achieves.

6.2. Compensations and transactions

Participants: Elena Giachino, Ivan Lanese, Fabrizio Montesi.

One of the predominant properties of CBUS is the loose coupling among the components. In fact, components can dynamically connect/disconnect and can be modified/updated at run time. It is thus important to support unexpected events, called faults. Typical fault handling mechanisms are normally programmed using try-catch operators. Nevertheless, the try-catch model does not seems appropriate to deal with low-level failures such as communication faults. At the application level, more advanced transactional models and primitives are needed to guarantee integrity and continuity of the whole system.

In [22] we describe the different issues that a language designer has to tackle when defining error handling mechanisms; we discuss how these issues can be studied in a model and then how they can be transported onto a full-fledged language. As a case study we use the language Jolie and the calculus SOCK it is based upon.

In [28] we have studied a global escape mechanism that can handle unexpected or unwanted conditions changing the default execution of communication sessions among distributed components, while preserving compatibility of the multiparty conversations.

In [45] we make a thorough comparison among different approaches to the specification of compensations, in particular static forms of recovery where the compensation is statically defined together with the transaction, and dynamic forms where the compensation is progressively built along with a computation. We have also proposed extensions of some of the approaches to handle nested transactions.

In [43] we investigate the notion of reversible computation in higher-order π-calculus. Reversibility is a main ingredient in the study of programming abstractions for reliable systems, e.g. for exception handling. In fact, reversibility can be used for going back to some consistent state after an exception has occurred. In [43] we have defined $\rho\pi$, an higher-order calculus where processes can both go forward and backward in the computation. This involves the definition of suitable memories for storing the past history of a process. We also defined a translation of $\rho\pi$ into a variant of higher-order π-calculus and proved its correctness.

6.3. Primitives for adaptable and evolvable components

Participants: Mario Bravetti, Ivan Lanese, Fabrizio Montesi, Jorge Perez Parra, Marco Pistore, Davide Sangiorgi, Gianluigi Zavattaro.

In Focus we study linguistic primitives for components, and models for them following the process calculus approach. A special emphasis is given to the adaptability and evolvability of the components — important issue in complex software systems. Components indeed are often used in contexts that had not been predicted at the time when the components were built. Moreover, the needs and the requirements on a system may change over time: one may find that the original specification was incomplete or ambiguous, or new needs may arise that had not been predicted at design time. As designing and deploying a system is costly, it is important that the system be capable of evolving and adapting itself to changes in the surrounding environment. Models and linguistic constructs for adaptability and evolvability of components are studied in [48], [52], [26].

In [41] we propose an approach to dynamic adaptation based on the combination of adaptation hooks provided by the adaptable application specifying where adaptation can happen, and adaptation rules external to the application, specifying when and how adaptation can be performed. This approach has been implemented on top of the language Jolie.

In [51] we propose linguistic primitives for a WEB OS middleware where components/programs (which can be active or passive, i.e., stored as a file) are addressed and invoked/executed via URLs. The idea is that, with respect to ordinary Operating Systems, addresses used to manage files and execute programs (via the basic load/execution mechanism) are extended from local filesystem path-names to global URLs. An immediate consequence is that, similarly as in traditional web technologies, executing a program at a given URL, can be done in two modalities: either the execution is performed client-side at the invoking machine (and relative URL addressing in the executed program set to refer to the invoked URL) or it is performed server-side at the machine addressed by the invoked URL (as, e.g., for a web service). The former mechanism provides a simple and widely-used mechanism to dynamically extend the set of components in execution in a local machine. A service update mechanism is also provided via primitives managing resources (e.g. passive components) by means of restful web services.

6.4. Verification of extensional properties

Participants: Cosimo Laneve, Marco Pistore, Davide Sangiorgi, Gianluigi Zavattaro.

“Extensional” refers to properties that have to do with behavioral descriptions of a system (i.e., how a system looks like from the outside). Examples of such properties include classical functional correctness, deadlock freedom, responsiveness (the fact that an answer to a query is eventually returned). Central issues in this study are compositionality and abstraction, as they are essential for mastering the complexity of modern software systems, especially the distributed ones. Compositionality allows one to symplify the analysis of a complex system by examining some subparts of it; and abstraction allows one to carry out the analysis on a more succint descriptions of the system, where only those aspects of a system that are relevant for the current analysis appear.

We mainly employ techniques based on behavioral equivalences (and preorders), and on types and logics. To be useful, the equivalence relations should also be congruences, i.e. preserved by all operators of the underlying languages. The equivalences we study are often coinductively defined (e.g., bisimulation [61]), whereas congruence is an inductive notion. Therefore this study also leads to studying the interplay between induction and coinduction. Related to equivalences is algebraic reasoning, which employs axioms and proof systems previously validated under the behavioural equality adopted.

Type systems offer a good trade-off between expressiveness and amenability to efficient verification and validation techniques.

In [17] we study how the termination property can facilitate and enhance the analysis of other properties, apparently quite different from termination itself. Specifically, we have considered the type-based techniques for termination of processes commented in the 'Resource Control' section, and the lock-freedom property. Intuitively, lock-freedom ensures that certain communications (or synchronizations) will eventually succeed. Lock-freedom is important in message-passing languages, where most liveness properties can be reduced to instances of lock-freedom. Examples are: Will the request of accessing a resource be eventually granted? Will the message sent by a client be eventually received from the server? And if so, will the server eventually send back an answer to the client? In multi-threaded programs, can a thread eventually acquire a lock? And if so, will the thread eventually release the lock? The lock-freedom property has also applications to other verification problems and program transformations, such as information flow analysis. Verification of liveness properties such as lock-freedom is notoriously hard in concurrency; it is particularly hard in presence of dynamic creation of threads and channels. In [17] we guarantee lock-freedom by means of a type system that is *hybrid*, in that it is obtained by appropriately combining type systems for termination and for deadlock-freedom. The resulting type system can type sophisticated recursive communication structures, and can be fully automated. Kobayashi has implemented a lock-freedom analysis tool based on the proposed type system, and tested for non-trivial programs.

In [39] we prove some new congruence result for the π-calculus with respect to bisimilarity. Specifically, we show that a particularly simple form of bisimilarity, which does not employ substitutions, is a congruence in certain dialects of the π-calculus.

In [24] we define a spatial logic for a general class of infinite-state transition systems, the Spatial Transition Systems (STS), where a monoidal structure on states accounts for the spatial dimension. We then show that the model checking problem for this logic is undecidable already when interpreted over Petri nets. Next, building on work by Finkel and Schnobelen, we introduce a subclass of STS, the Well-Structured STS (WSSTS), which is general enough to include models such as Petri nets, Broadcast Protocols, CCS and Weighted Automata. Over WSSTS, an interesting fragment of spatial logic — the monotone fragment — turns out to be decidable under reasonable effectiveness assumptions. For this class of systems, we also offer a Hennessy-Milner theorem, characterizing the logical preorder induced by the monotone fragment as the largest spatial-behavioural simulation. We finally prove that, differently from the corresponding logic, this preorder is in general not decidable, even when confining to effective WSSTS.

The paper [37] studies the enhancement of monitoring techniques for services with formal verification techniques. The approach explicitly encodes assumptions that the constituent services of a service-based applications will perform as expected. The goal of the approach is to be able to (a) pro-actively deciding whether the requirements of the applications will be violated based on monitored failures, and (b) identifying the specific root cause for the violated requirements.

In [46] (in a collaboration with Magneti-Marelli Powertrain) we study an extensional property that is useful in analysis of codes of control switchboards. The property relates intervals of values of input variables to intervals of values of output variables of programs. The technique we use is known in the literature as interval analysis. We demonstrate that the interval of multilinear expressions may be computed precisely studying their values at the bounds of the variables therein. (Multilinear expressions have factors using different variables.) We then propose a technique for the interval analysis of generic expressions that transforms them into multilinear ones and computes the interval of the latters. Overall, our technique gives a better precision since non-multilinear expressions are quite infrequent in programs. An open issue is the study of interval analysis in multithreaded programs with shared variables.

Sangiorgi has nearly completed a book on induction and coinduction.

6.5. Resource Control

Participants: Romain Demangeon, Ugo Dal Lago, Marco Gaboardi, Daniel Hirschkoff, Simone Martini, Davide Sangiorgi, Paolo Parisen Toldin.

In Focus, we study methodologies and foundational tools for the analysis of resource control in programming languages and processes. The techniques employed come from the world of type theory and proof theory, and as such have been used extensively in the context of sequential computation. Interesting results have been obtained recently indicating that those techniques can be quite useful in the concurrent context too, thus being potentially interesting for CBUS. Results in this direction have been published in [34], [25], [35], [36], [55], [32], [29], [14], [15], [50]

In [34], techniques ensuring termination combining tools coming from term rewriting with logical relations are introduced and exploited in the context of a π-calculus with functional and imperative names. Noticeably, the introduced technique allows to capture a large class of programs and processes. This, and other techniques, are collected in Demangeon's PhD thesis [10].

Techniques inspired by linear logic have been investigated extensively. In [25], a functional programming language inspired by light linear logic has been introduced and proved sound and complete with respect to the class of polynomial time computable functions. The class of problems decideable in polynomial space can be itself characterized by a language inspired by a subsystem of linear logic, namely soft linear logic [55]. These quantitative properties remain true if reduction is performed optimally [50].

Bidirectional computation is another computation paradigm where techniques coming from linear logic can find useful applications, this time when defining functional programming languages with sub-linear space consumption [32].

We can go even further, beyond classical, sequential computation: higher-order processes can be linearized, and this linearization makes it possible to isolate a class of processes which terminates in polynomial time on their size [30]. Quantum polynomial time can be characterized similarly [15].

One of the main problems affecting systems defined this way is their poor intentional expressive power. Some investigations have been directed towards the definition of systems capturing natural algorithms. One result of this investigation has been the definition of quantified bounded affine logic, which has been proved to be at least as (intensionally) expressive as different, heterogeneous existing systems [53].

More semantical investigations have been also carried out. Realizability models have been shown to be a useful tool in the study of quantitative properties of programs [14]. Linearity has been analyzed semantically in the context of categorical and coherent models [56]. The use of quasi-interpretation techniques to study quantitative properties of stream programs have been pursued in [35].

Finally, some results have been obtained [29] in the context of rewriting-based techniques.

6.6. Semantics of ad-hoc networks

Participants: Ivan Lanese, Davide Sangiorgi, Gianluigi Zavattaro.

By ad hoc networks we refer to networks of devices connected by wireless links and comminicating via broadcast. The devices may also be mobile.

In [33] we consider a model of these systems recently proposed by Singh, Ramakrishnan, and Smolka, and verification problems for it that can be expressed as reachability of configurations with one node (resp. all nodes) in a certain state. All decision problems are parametric both on the size and on the form of the communication topology of the initial configurations. We draw a complete picture of the decidability boundaries of these problems according to various assumptions on the communication topology of the network, namely static vs mobile and unbounded- vs bounded-path topologies.

In [19] we develop an operational semantics for a calculus of wireless systems. We present different Reduction Semantics and a Labelled Transition Semantics and prove correspondence results between them. A major goal of the semantics is to describe the forms of interference among the activities of processes that are peculiar of wireless systems. Such interference occurs when a location is simultaneously reached by two transmissions. We use the calculus to describe and analyse a few properties of a version of the Alternating Bit Protocol.

6.7. Expressiveness of computational models

Participants: Maurizio Gabbrielli, Cosimo Laneve, Ivan Lanese, Jacopo Mauro, Jorge Perez Parra, Davide Sangiorgi, Gianluigi Zavattaro.

Expressiveness refers to the study of the expressive power of computational models. In Focus, this topic involves two main aspects. First, there may be several candidate models for describing a specific property or system. A formal comparison among the models is then needed to establish which one should be preferred. The choice of the model can have important consequences on practical issues such as verification; for instance some properties may be decidable in a model and not in another one.

A second aspect in the study of expressiveness in Focus concerns the comparison of models used at different levels of description of a system. Here it is important to understand whether a model implements another one, if a model refines another, and so on.

The Focus members have a solid experience in the general theme of expressiveness of foundational computational models. and maintain interest in the topic. Therefore, whenever we find it appropriate we investigate questions of comparison among models, and the transfer of concepts and techniques among different computational models, including models that go beyond CBUS (e.g., models for biological systems). We briefly review below the work carried out in the past year.

A synchronization is a mechanism allowing two or more processes to perform actions at the same time. In [47], we study the expressive power of synchronization primitives in relation to the number of processes involved. One of the main results is the nonexistence of a uniform, fully distributed translation of languages in which synchronizations involve $n + 1$ processes into languages with synchronizations of n processes.

Higher-order process calculi are formalisms for concurrency in which processes can be passed around in communications. Higher-order (or process-passing) concurrency is often presented as an alternative paradigm to the first order (or name-passing) concurrency of the π-calculus for the description of mobile systems. These calculi are inspired by, and formally close to, the λ-calculus, whose basic computational step — beta-reduction — involves term instantiation. The theory of higher-order process calculi is more complex than that of first-order process calculi. This shows up in, for instance, the definition of behavioral equivalences. A long-standing approach to overcome this burden is to define encodings of higher-order processes into a first-order setting, so as to transfer the theory of the first-order paradigm to the higher-order one. While satisfactory in the case of calculi with basic (higher-order) primitives, this indirect approach falls short in the case of higher-order process calculi featuring constructs for, e.g., localities and dynamic system reconfiguration, which are frequent in modern distributed systems. Indeed, for higher-order process calculi involving little more than traditional process communication, encodings into some first-order language are difficult to handle or do not exist. The works [11], [20], [44], [18] offer foundational studies for higher-order process calculi. We concentrate on two closely interwoven issues in process calculi: expressiveness and decidability. Surprisingly, these issues have been little explored in the higher-order setting. Our research is centered around a core calculus for higher-order concurrency in which only the operators strictly necessary to obtain higher-order communication are retained. We develop the basic theory of this core calculus and rely on it to study the expressive power of basic features of concurrent languages, namely synchrony, forwarding, and polyadic communication (the possibility of exchanging more than one value in a single communication step).

In [16] we study the decidability of termination for two CHR dialects. CHR is a constraint, rule based concurrent language whose peculiarity is to allow multiple atomic formulas in the head of rules. Such a feature augments the expressive power of the languages, as proved elsewhere (somehow, it can be considered similar to multiple ynchronization gathering n processes simultaneously). In this paper we consider a CHR language that do not allow function symbols (of arity > 0) and allow the use of the = built-in in the body of rules. This language is as expressive as Turing Machines. We then consider a first dialect where variables in the body of a rule must appear in the head. We show that the existence of an infinite computation is decidable for this dialect. Next we consider a second dialect which instead limits the number of atoms in the head of rules to one. We prove that, in this case, the existence of a terminating computation is decidable. These results show that both dialects are strictly less expressive than Turing Machines.

In [21] we compare and contrast (also by means of encodings) a well-established concurrency model such as the π-calculus, with an interesting variant of λ-calculus, namely Curien and Herbelin's $\overline{\lambda}\mu\widetilde{\mu}$. This calculus, roughly a λ-calculus equipped with control operators, has significant differences with respect to λ-calculus: it has a striking dualism, that reminds of the dualism between input and output in concurrency, it is non-confluent (as most concurrency models), and allows a Curry-Howard isomorphism with the classical sequent calculus.

In [13] we explore the expressive power of languages that naturally model biochemical interactions with respect to languages that naturally model only basic chemical reactions, identifying molecular association as the basic mechanism that distinguishes the former from the latter. We use a process algebra, the Biochemical Ground Form (BGF), which extends with primitives for molecular association CGF, a process algebra proved to be equivalent to the traditional notations for describing basic chemical reactions. We first observe that, differently from CGF, BGF is Turing universal as it supports a finite precise encoding of Random Access Machines, a well-known Turing powerful formalism. Then we prove that the Turing universality of BGF derives from the interplay between the molecular primitives of association and dissociation. In fact, the elimination from BGF of the primitives already present in CGF does not reduce the computational strength of the process algebra, while if either association or dissociation is removed then BGF is no longer Turing complete.

7. Contracts and Grants with Industry

7.1. Contracts with Industry

- Contract with the Methodologies & Tools team of Magneti Marelli Powertrain S.p.A., Bologna, Italy, for developing interval analysis techniques of multithreaded programs in automotive control systems. (Contract followed by C. Laneve.)

7.2. Grants with Industry

- Fellowship to Tudor Lascu from July 2010 to December 2010 from Magneti Marelli Powertrain S.p.A.

7.3. National Initiatives

- AEOLUS (Mastering the Cloud Complexity) is an ANR-ARPEGE project started on 1st December 2010 and with a 40-month duration. AEOLUS studies the problem of installation, maintenance and update of package-based software distributions in cloud-based distributed systems. The problem consists of representing the distribution and the dependencies of packages, in such a way that *modification plans* can be (semi)automatically synthesized when packages should be updated or the system should be reconfigured. Main persons involved: Sangiorgi, Zavattaro.

- S. Martini, U. Dal Lago and M. Gaboardi are involved in the CNRS PICS 2010 (“International Projects for Scientific Cooperation”) project “Linear Logic and applications”.
- Members of Focus are involved in a few small italian projects (about 10KE per year): M. Bravetti and D. Sangiorgi in the project “Processes in Open-ended Distributed Systems”, M. Gabbrielli in the project “Innovative and multi-disciplinary approaches for constraint and preference reasoning”; S. Martini, U. Dal Lago and M. Gaboardi in the project “CONtrollo e CERTificazione dell’usO delle risorse”.

7.4. European Initiatives

- Sensoria (Software Engineering for Service-Oriented Overlay Computers) is an EU Integrated Project from FP6, started in 2005 and ended in April 2010. Sensoria has studied theories, techniques and methods for service-oriented architectures integrated in a pragmatic software engineering approach. Main Focus members involved: Bravetti, Lanese, Sangiorgi, Zavattaro.
- Hats (Highly Adaptable and Trustworthy Software using Formal Models) is an EU Integrated Project from FP7, started March 2009 and with a 4 year duration. Hats studies formal methods for obtaining high adaptability combined with trustworthiness in the setting of object-oriented languages and software product lines. Most Focus members are involved.

7.5. International Initiatives

We list here the cooperations and contacts with other groups.

- Inria EPI Sardes (on models and languages for components). Contact person(s) in Focus: Lanese, Perez, Sangiorgi. A number of visits in both directions. One joint PhD supervision (C. Mezzina).
- ENS Lyon (on concurrency models and resource control). Contact person(s) in Focus: Dal Lago, Martini, Sangiorgi. Several visit exchanges during the year, in both directions. One joint PhD supervision (R. Demangeon).
- Laboratoire d’Informatique – Université Paris Nord, Villetaneuse (on implicit computational complexity). Contact person(s) in Focus: Martini. Martini has been professeur invité for one month in November-December 2010.
- INRIA-Saclay and Ecole Polytechnique (on formal languages for mobile, secure, stochastic and distributed systems). Contact person(s) in Focus: Gabbrielli. Gabbrielli has been invited researcher at INRIA-Saclay and Ecole Polytechnique from June 15 to August 15, 2010, in the Inria Team Comete.
- University of Paris-Diderot (Paris 7) (on expressiveness of computational models). Contact person(s) in Focus: Gabbrielli. There is an agreement between the University of Bologna and the University of Paris-Diderot (Paris 7) for a joint thesis sponsorship concerning the PhD student Ferdinanda Camporesi (advisors Maurizio Gabbrielli and Radhia Cousot).
- University of Paris-Diderot Paris 7, Team PPS (on logics for processes and resource control). Contact person(s) in Focus: Dal Lago, Martini, Sangiorgi, Zavattaro. Various short visits, including one-week visit from T. Ehrhard.
- INRIA-MICROSOFT Research Center of Paris (on models of computation). Contact person(s) in Focus: Laneve. Laneve has been invited scientist in June 2010 (1 month).
- University of Trento and FBK Trento (on service-oriented computing). Contact person(s) in Focus: Lanese, Sangiorgi. Short visits in both directions.
- Icelanding Center of Excellence in Theoretical Computer Science, Reykjavik (on logics for processes). Contact person(s) in Focus: Sangiorgi. One visit exchange in 2010.
- Cambridge Microsoft Research Center (on Models of computation). Contact person(s) in Focus: Laneve. Laneve has been invited scientist in January 2010 (1 month).

- School of Information Sciences, Tohoku University, Japan (on type systems and coinductive techniques for concurrent languages). Contact person(s) in Focus: Sangiorgi. Visit exchanges in previous years; completion of work continued during 2010.
- Facultad de Informatica, Universidad Complutense de Madrid (on web services). Contact person(s) in Focus: Bravetti. Bravetti is an external collaborator in the Spanish Ministry of Science and Education project TESIS (advanced methodologies and tools for TESting and web servIceS).

7.6. Exterior research visitors

- Alessio Guglielmi, University of Bath and LORIA and INRIA Nancy-Grand Est, 19-21 January. Topic: Normalisation and Proof Nets.
- Laura Bocchi, University of Leicester, 17-19 February. Topic: sessions and contracts.
- Alan Schmitt, Inria Grenoble, 1 week in February. Topic: models for components.
- Claudio Mezzina, Inria Grenoble, 2 1-month visits, in May and November . Topic: constructs for reversible computations.
- Romain Péchoux, maître de conférence de l'Université Nancy 2, Nancy-Université, 12-17 April 2010 and 18-24 July 2010. Topic: Space properties of stream programs.
- Stefania Gnesi (ISTI-CNR), and Alessandro Fantechi (University Florence): 28-29 April. Topic: Models for Software Product Lines.
- Matteo Cimini, Icelanding Center of Excellence in Theoretical Computer Science, Reykjavik, 1 week in April. Topic: logics for concurrent languages.
- Colin Riba, maître de conférence chaire CNRS de l'École Normale Supérieur de Lyon. Period: 23-27 Mai 2010. Topic: Realizability and Gödel's Dialectica Interpretation.
- Yannick Welsch, University of Kaiserlautern, 1 week in May. Topic: components and objects.
- Brian Redmond, postdoc at the University of Calgary. Period: 13-16 July 2010. Topic: A logic for PSPACE.
- Ulrich Schöpp, postdoc at LMU, Munich. Period: 6-10 September 2010. Topic: functional programming in sublinear space.
- Claudia Faggian, Chargé de Recherce au CNRS, PPS. Period: 10-17 November 2010. Topic: quantum computation and linear logic.
- Immo Grabe, PhD student at CWI, Amsterdam, The Netherlands. Period: 18-21 October 2010. Topic: Deadlock detection for active objects.
- Marco Patrignani, University Leuven. 2 weeks in October. Topic: Types for objects and processes.
- Paolo di Giamberardino, ATER de l'Université Paris 13. Period: 25-27 October. Topic: session types and soft linear logic.
- Lucian Wishick, Microsoft Redmond: 15-16 November 2010. Topic: Programming languages for distributed systems.
- Roberto Zunino, Università di Trento, 23-24 November 2010. Topic: sessions and contracts.

8. Dissemination

8.1. Animation of the scientific community

M. Bravetti: Program Committee co-Chair of the 7th International Workshop on Web Services and Formal Methods: formal aspects of service oriented and cloud computing (WS-FM 2010). Program Committee member of the following conferences: 16th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2010), 4th International Workshop on Verification and Evaluation of Computer and Communication Systems (VECoS 2010), 3rd International Conference on Software Testing, Verification, and Validation (ICST 2010), 3rd IEEE International Conference on Cloud Computing (CLOUD 2010), IFIP International Conference on Formal Techniques for Distributed Systems (FMOODS/FORTE 2010), 5th European Young Researchers Workshop on Service-Oriented Computing (YR-SOC 2010).

U. Dal Lago: Invited speaker at the 11th Workshop on Logic and Computational Complexity (LCC 2010), Edinburgh, UK; July 10th, 2010. Program Committee member of the following conferences: 37th International Colloquium on Automata, Languages and Programming (ICALP 2010), Bordeaux, France, July 5-10 2010; First International Workshop on Developments in Implicit Computational Complexity (DICE 2010), 26-28 March 2010, Paphos, Cyprus; First International Symposium on Games, Automata, Logics and Formal Verification (GANDALF 2010), 17-18 June 2010, Minori, Italy; Tenth International Conference on Typed Lambda Calculi and Applications (TLCA 2011), Novisad, Serbia, June 1 - 3, 2011.
I. Lanese: Program Committee member of 3nd Interaction and Concurrency Experience (ICE 2010), Amsterdam, Netherlands, June 10, 2010. Track Chair of the Service Oriented Architectures and Programming track of the 25th Annual ACM Symposium on Applied Computing (SAC 2010), Sierre, Switzerland, March 22-26, 2010.
C. Laneve: Program Committee member of: Structural Operational Semantics 2010 (SOS), August 30, 2010, Paris, France.
M. Lienhardt:. Invited presentation at the conference "Formal Methods for Components and Objects", Gratz, Austria, November 2010.
M. Gabbrielli: Member of the editorial board of the journal TPLP (Theory and Practice of Logic Programming). Program Committee member of: ICLP 2010 - Twenty-sixth International Conference on Logic Programming. Program Committee member of: FSEN '11, 4th International Conference on Fundamentals of Software Engineering, 2011.
S. Martini: Invited speaker at Developments in Implicit computational complexity (DICE 2010), Paphos, Cyprus; March 27-28, 2010. Invited speaker at Logique et Interaction : vers une Geometrie de la Cognition (LIGC 2010), Paris; November 18-20, 2010. Member of the final jury for the PhD of Etienne Duchesne, Université de Marseille (Institut de Mathématiques de Luminy). Member of the final jury for the PhD of Paolo Di Benedetto and Carlo Rosa, Università dell'Aquila (Italy). Editor of a special issue of ACM Transaction on Computational Logic, Vol. 11, No. 4, with selected papers of Computer Science Logic 2008.
D. Sangiorgi: Member of the editorial board of the journals: Logical Methods in Computer Science, Acta Informatica, and Distributed Computing. Chairman of the IFIP Working Group 2.2 (Formal description of programming concepts; it has recently celebrated the 40th year of activity). Program Committee member of the conferences: Tenth International Workshop on Coalgebraic Methods in Computer Science 26-28 March 2010, Paphos, Cyprus; the Twenty-sixth Conference on the Mathematical Foundations of Programming Semantics, Ottawa, Ontario, Canada May 6 - 10, 2010; 35th International Symposium on Mathematical Foundations of Computer Science, Brno, Czech Republic, 21-27 August 2010. Invited speaker at the fifth Symposium on Trustworthy Global Computing, Munich (Germany), Ferbruary 24-26, 2010.
G. Zavattaro: Member of the IFIP (International Federation for Information Processing) Working Group 6.1 "Architectures and Protocols for Distributed Systems", of the editorial board of the "Journal of Software (JSW)", and of the steering committee of the "International Conference on Coordination Models and Languages (COORDINATION)". Invited speaker during the celebration of the 30th anniversary of the conference FORTE (Amsterdam, June 2010). Publicity chair for the joint conference on Distributed Computing Techniques (DisCoTec, 2010). Program Committee member of the conferences: 8th IEEE European Conference on Web Services (ECOWS'10), Ayia Napa, Cyprus, 1–3 December, 2008; 1st International Conference on Cloud Computing, GRIDs, and Virtualization (CLOUD COMPUTING 2010), Lisbon, Portugal, 21-26 November, 2010; 7th International Workshop on Web Services and Formal Methods (WS-FM'10), Hoboken, New Jersey, USA, 16-17 September, 2010; 8th International Conference on Software Engineering and Formal Methods (SEFM'10). Pisa, Italy, 13–17 September, 2010; 17th Int. Workshop on Expressiveness in Concurrency (EXPRESS'10), Paris, France, 30 August, 2010; 4th IEEE International Symposium on Theoretical Aspects of Software Engineering (TASE'10), Taipei, Taiwan, 25–27 August 2010; 4th Membrane Computing and Biologically Inspired Process Calculi workshop (MeCBIC'10), Jena, Germany, 23-24 August, 2010; 7th IEEE International Conference on Services Computing (SCC'10) and 3rd IEEE International Conference on Cloud Computing (CLOUD'10), Miami, Florida, USA, 5–10 July, 2010; 1st Int. Workshop on Interactions between Computer Science and Biology (CS2Bio'10), Amsterdam, The Netherlands, 10 June 2010; Cloud Computing and Virtualization Academic Conference (CCV'10), Singapore, 17-18 May, 2010.

8.2. Other initiatives

- M. Bravetti and G. Zavattaro founded in 2004 the international workshop WS-FM on Web Services and Formal Methods, which has been held, since then, every year in sites all around the world (Pisa, Versailles, Wien, Brisbane, Milan, Bologna and New Jersey) and which helped in forming a community of researchers working in the area of service oriented computing and formal methods.

8.3. Teaching

M. Bravetti: 92 hours of teaching at the University of Bologna (an undergraduate course on Web Technologies and a graduate course on Advanced Web Technologies).
U. Dal Lago: 130 hours of teaching at the University of Bologna (a course on Cryptography, part of a course on Programming in Java, a course on Programming in Python for students in bioinformatics, Quantum Computation for graduate students). A 10-hour post-graduate course at the ESSLLI international Summer School, Copenhagen, Denmark, 9-20 August, 2010.
M. Gabbrielli: 100 hours of teaching at the University of Bologna (a course on Programming Languages, a course on Artificial Intelligence).
I. Lanese: A 4-hours post-graduate course at the 1st ADAPT Summer School, Koblenz, Germany, 26 September-2 October 2010.
C. Laneve: 100 hours of teaching at University of Bologna (a course on Program Analysis).
S. Martini: 100 hours of teaching at the University of Bologna (a course on Programming Languages, a course on Types in programming languages, a short module on Computer abilities for biologists).
D. Sangiorgi: 120 hours of teaching at the University of Bologna (a course on Operating systems, and a short module on models for concurrency). A 20-hour post-graduate course at the BISS international Spring School, Bertinoro, Italy, 1-12 March 2010.
G. Zavattaro: 120 hours of teaching at the University of Bologna, Cesena campus (a course on Theoretical Computer Science and a course on Programming Languages).

8.4. Completed Phd thesis

- Jorge A. Perez, Higher-Order Concurrency: Expressiveness and Decidability Results. Dottorato di Ricerca in Informatica, University of Bologna, April 2010. Supervisors: Sangiorgi
- A. Vitale, Expressiveness in biologically inspired language. Dottorato di Ricerca in Informatica, University of Bologna, April 2010. Supervisor: Laneve
- R. Demangeon, Termination for Concurrent Languages, PhD thesis, Ecole Normale Sup rieure de Lyon, Nov. 2010. Supervisor: Hirschkoff, Sangiorgi.

9. 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, p. 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, p. 1191-1222.

[3] S. CARPINETI, C. LANEVE, L. PADOVANI. *PiDuce - a project for experimenting Web services technologies*, in "Science of Computer Programming", 2009, vol. 74, n^o 10, p. 777–811.

[4] 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, p. 219-260.

[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. HOFMANN. *Quantitative Models and Implicit Complexity*, in "FSTTCS 2005: Foundations of Software Technology and Theoretical Computer Science, 25th International Conference", R. RAMANUJAM, S. SEN (editors), Lecture Notes in Computer Science, Springer, 2005, vol. 3821, p. 189-200.

[7] F. MONTESI, C. GUIDI, G. ZAVATTARO. *Composing Services with JOLIE*, in "Fifth IEEE European Conference on Web Services (ECOWS 2007)", 2007, p. 13-22.

[8] B. PIERCE, D. SANGIORGI. *Behavioral equivalence in the polymorphic pi-calculus*, in "Journal of the ACM", 2000, vol. 47, n^{o} 3, p. 531–584.

[9] D. SANGIORGI, D. WALKER. *The π-calculus: a Theory of Mobile Processes*, Cambridge University Press, 2001.

Publications of the year

Doctoral Dissertations and Habilitation Theses

[10] R. DEMANGEON. *Termination for Concurrent Languages*, Ecole Normale Supérieure de Lyon and University of Bologna, 2010.

[11] J. A. PEREZ. *Higher-Order Concurrency: Expressiveness and Decidability Results*, University of Bologna, 2010.

[12] A. VITALE. *Expressiveness in biologically inspired language*, University of Bologna, 2010.

Articles in International Peer-Reviewed Journal

[13] L. CARDELLI, G. ZAVATTARO. *Turing universality of the Biochemical Ground Form*, in "Mathematical Structures in Computer Science", 2010, vol. 20, n^{o} 1, p. 45–73, http://dx.doi.org/10.1017/S0960129509990259.

[14] U. DAL LAGO, M. HOFMANN. *A Semantic Proof of Polytime Soundness for Light Affine Logic*, in "Theory of Computing Systems", 2010, vol. 46, n^{o} 4, p. 673-689, http://dx.doi.org/10.1007/s00224-009-9210-x.

[15] U. DAL LAGO, A. MASINI, M. ZORZI. *Quantum Implicit Computational Complexity*, in "Theoretical Computer Science", 2010, vol. 411, n^{o} 2, p. 377-409, http://dx.doi.org/10.1016/j.tcs.2009.07.045.

[16] M. GABBRIELLI, J. MAURO, M. C. MEO, J. SNEYERS. *Decidability properties for fragments of CHR*, in "TPLP", 2010, vol. 10, n^{o} 4-6, p. 611-626, http://arxiv.org/abs/1007.4476.

[17] N. KOBAYASHI, D. SANGIORGI. *A hybrid type system for lock-freedom of mobile processes*, in "ACM Trans. Program. Lang. Syst.", 2010, vol. 32, n^{o} 5.

[18] I. LANESE, J. A. PEREZ, D. SANGIORGI, A. SCHMITT. *On the Expressiveness and Decidability of Higher-Order Process Calculi*, in "Inf. Comput.", 2010, To appear.

[19] I. LANESE, D. SANGIORGI. *An operational semantics for a calculus for wireless systems*, in "Theor. Comput. Sci.", 2010, vol. 411, n^{o} 19, p. 1928–1948.

Articles in Non Peer-Reviewed Journal

[20] J. A. PEREZ. *Higher-Order Concurrency: Expressiveness and Decidability Results – A Survey*, in "Bulletin of the EATCS", June 2010, vol. 101, p. 92–124.

Invited Conferences

[21] M. CIMINI, C. S. COEN, D. SANGIORGI. *Functions as processes: termination and the lambda mu mu - calculus*, in "Proceedings of TGC 2010, 5th International Symposium on Trustworty Global Computing", A. R. MARTIN HOFMANN (editor), LNCS, Springer, 2010, vol. 6084, p. 73–86, http://www.cs.unibo.it/~sangio/DOC_public/lambdamu.pdf.

[22] I. LANESE, F. MONTESI. *Error Handling: From Theory to Practice*, in "Proceedings of ISOLA 2010, 4th International Symposium On Leveraging Applications of Formal Methods, Verification and Validation", T. MARGARIA, B. STEFFEN (editors), LNCS, Springer, 2010, vol. 6416, p. 66-81, http://www.cs.unibo.it/~lanese/publications/fulltext/isola2010.pdf.gz.

International Peer-Reviewed Conference/Proceedings

[23] L. ACCIAI, M. BOREALE, G. ZAVATTARO. *Behavioural Contracts with Request-Response Operations*, in "COORDINATION", Lecture Notes in Computer Science, Springer, 2010, vol. 6116, p. 16-30, http://dx.doi.org/10.1007/978-3-642-13414-2_2.

[24] L. ACCIAI, M. BOREALE, G. ZAVATTARO. *On the Relationship between Spatial Logics and Behavioral Simulations*, in "FOSSACS", Lecture Notes in Computer Science, Springer, 2010, vol. 6014, p. 146-160, http://dx.doi.org/10.1007/978-3-642-12032-9_11.

[25] P. BAILLOT, M. GABOARDI, V. MOGBIL. *A PolyTime Functional Language from Light Linear Logic*, in "ESOP 2010", LNCS, Springer, 2010, vol. 6012, p. 104-124, http://www.cs.unibo.it/~gaboardi/publication/BaillotGaboardiMogbil10esop.pdf.

[26] M. BRAVETTI, C. D. GIUSTO, J. A. PEREZ, G. ZAVATTARO. *A Calculus for Component Evolvability (Extended Abstract)*, in "Formal Aspects of Component Software (FACS 2010)", Lecture Notes in Computer Science, Springer, 2010, to appear, http://www.cs.unibo.it/~bravetti/papers/facs10.pdf.

[27] R. BRUNI, A. KERSTEN, I. LANESE, G. SPAGNOLO. *A new Strategy for Distributed Compensations with Interruption in Long-Running Transactions*, in "Proceedings of WADT 2010", 2010, submitted.

[28] S. CAPECCHI, E. GIACHINO, N. YOSHIDA. *Global Escape in Multiparty Sessions*, in "Proceedings of the 30th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'10)", 2010, to appear.

[29] U. DAL LAGO, S. MARTINI. *Derivational Complexity is an Invariant Cost Model*, in "Foundational and Practical Aspects of Resource Analysis, FOPARA 2009", LNCS, 2010, vol. 6324, p. 88 – 101, http://www.cs.unibo.it/~martini/papers-to-ftp/dcicm.pdf.

[30] U. DAL LAGO, S. MARTINI, D. SANGIORGI. *Light Logics and Higher-Order Processes*, in "17th International Workshop on Expressiveness in Concurrency, EXPRESS 2010", EPTCS, 2010, To appear.

[31] U. DAL LAGO, S. MARTINI, M. ZORZI. *General Ramified Recurrence is Sound for Polynomial Time*, in "Developments in Implicit Computational complexity, DICE 2010", EPTCS, 2010, vol. 23, p. 47–62, http://arxiv.org/abs/1005.0521v1.

[32] U. DAL LAGO, U. SCHÖEPP. *Functional Programming in Sublinear Space*, in "Programming Languages and Systems, 19th European Symposium, Proceedings", Lecture Notes in Computer Science, Springer, 2010, vol. 6012, p. 205-225.

[33] G. DELZANNO, A. SANGNIER, G. ZAVATTARO. *Parameterized Verification of Ad Hoc Networks*, in "CONCUR", Lecture Notes in Computer Science, Springer, 2010, vol. 6269, p. 313-327, http://dx.doi.org/10.1007/978-3-642-15375-4_22.

[34] R. DEMANGEON, D. HIRSCHKOFF, D. SANGIORGI. *Termination in Impure Concurrent Languages*, in "Proc. of CONCUR", LNCS, Springer Verlag, 2010, vol. 6269, p. 328–342, http://perso.ens-lyon.fr/romain.demangeon/Recherche/impfun.pdf.

[35] M. GABOARDI, R. PÉCHOUX. *Global and Local Space Properties of Stream Programs*, in "FOPARA'09", LNCS, 2010, vol. 6324, p. 51 – 66, http://www.cs.unibo.it/~gaboardi/publication/GaboardiPechoux10fopara.pdf.

[36] M. GABOARDI, M. PICCOLO. *Categorical Models for a Semantically Linear λ-calculus*, in "LINEARITY'09", EPTCS, 2010, vol. 22, p. 1 – 13, http://www.cs.unibo.it/~gaboardi/publication/GaboardiPiccolo10linearity.pdf.

[37] A. GEHLERT, A. BUCCHIARONE, R. KAZHAMIAKIN, A. METZGER, M. PISTORE, K. POHL. *Exploiting assumption-based verification for the adaptation of service-based applications*, in "Proc. ACM Symposium on Applied Computing (SAC)", ACM, 2010, p. 2430–2437.

[38] E. GIACHINO. *On Semantic Subtyping and Safe Object-Oriented Sessions*, in "Proceedings of the 5th Workshop on Intersection Types and Related Systems (ITRS'10)", 2010, to appear.

[39] D. HIRSCHKOFF, D. POUS. *On Bisimilarity and Substitution in Presence of Replication*, in "Proc. ICALP", LNCS, Springer Verlag, 2010, vol. 6199, p. 454–465, http://hal.archives-ouvertes.fr/docs/00/49/17/66/PDF/seeds.pdf.

[40] R. KAZHAMIAKIN, V. KERHET, M. PAOLUCCI, M. PISTORE, M. WAGNER. *Use-Based Discovery of Pervasive Services*, in "14th Int. Conf. on Artificial Intelligence: Methodology, Systems, and Applications (AIMSA)", Lecture Notes in Computer Science, Springer, 2010, vol. 6304, p. 11–20.

[41] I. LANESE, A. BUCCHIARONE, F. MONTESI. *A Framework for Rule-based Dynamic Adaptation*, in "Proceedings of TGC 2010, 5th International Symposium on Trustworty Global Computing", A. R. MARTIN HOF-

MANN (editor), LNCS, Springer, 2010, vol. 6084, p. 284–300, http://www.cs.unibo.it/~lanese/publications/fulltext/tgc2010.pdf.gz.

[42] I. LANESE. *Static vs Dynamic SAGAs*, in "Proceedings of ICE 2010, 3rd Interaction and Concurrency Experience", S. BLIUDZE, R. BRUNI, D. GROHMANN, A. SILVA (editors), Electronic Proceedings in Theoretical Computer Science, 2010, vol. 38, p. 51–65, http://www.cs.unibo.it/~lanese/publications/fulltext/ice2010.pdf.gz.

[43] I. LANESE, C. A. MEZZINA, J.-B. STEFANI. *Reversing Higher-Order Pi*, in "CONCUR", P. GASTIN, F. LAROUSSINIE (editors), Lecture Notes in Computer Science, Springer, 2010, vol. 6269, p. 478–493, http://www.cs.unibo.it/~lanese/publications/fulltext/concur2010.pdf.gz.

[44] I. LANESE, J. A. PEREZ, D. SANGIORGI, A. SCHMITT. *On the Expressiveness of Polyadic and Synchronous Communication in Higher-Order Process Calculi*, in "Proc. of ICALP'10", Lecture Notes in Computer Science, Springer, 2010, vol. 6199, p. 442–453.

[45] I. LANESE, C. VAZ, C. FERREIRA. *On the Expressive Power of Primitives for Compensation Handling*, in "ESOP", A. D. GORDON (editor), Lecture Notes in Computer Science, Springer, 2010, vol. 6012, p. 366–386, http://www.cs.unibo.it/~lanese/publications/fulltext/esop2010.pdf.gz.

[46] C. LANEVE, T. A. LASCU, V. SORDONI. *The Interval Analysis of Multilinear Expressions*, in "Proc. Tools for Automatic Program AnalysiS (TAPAS)", October 2010, vol. 267, p. 43–53.

[47] C. LANEVE, A. VITALE. *The Expressive Power of Synchronizations*, in "Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010", IEEE Computer Society, 2010, p. 382–391.

[48] F. MONTESI, D. SANGIORGI. *A model of evolvable components*, in "Proceedings of TGC 2010, 5th International Symposium on Trustworty Global Computing", A. R. MARTIN HOFMANN (editor), LNCS, Springer, 2010, vol. 6084, p. 153–171, http://www.cs.unibo.it/~sangio/DOC_public/evolv.pdf.

Scientific Books (or Scientific Book chapters)

[49] M. GABBRIELLI, S. MARTINI. *Programming Languages: Principles and Paradigms*, Springer, 2010.

Other Publications

[50] P. BAILLOT, P. COPPOLA, U. DAL LAGO. *Light Logics and Optimal Reduction: Completeness and Complexity*, 2010, Submitted.

[51] M. BRAVETTI. *File Managing and Program Execution in Web Operating Systems*, 2010, Submitted, http://arxiv.org/abs/1005.5045.

[52] M. BRAVETTI, C. D. GIUSTO, J. A. PEREZ, G. ZAVATTARO. *Bounded and Eventual Adaptation for Evolvable Components*, 2010, submitted, http://www.cs.unibo.it/~bravetti/papers/esop11.pdf.

[53] U. DAL LAGO, M. HOFMANN. *Bounded Linear Logic, Revisited*, 2010, Submitted.

[54] M. GABBRIELLI, C. GUIDI, J. MAURO, F. MONTESI. *An efficient management of correlation sets*, 2010, Submitted.

[55] M. GABOARDI, J.-Y. MARION, S. R. D. ROCCA. *An implicit characterization of PSPACE*, 2010, Submitted, http://www.cs.unibo.it/~gaboardi/publication/GaboardiMarionRonchi10pspace.pdf.

[56] M. GABOARDI, L. PAOLINI, M. PICCOLO. *Full Abstraction for a Linear PCF*, 2010, Submitted, http://www.cs.unibo.it/~gaboardi/publication/GaboardiPaoliniPiccolo10slpcf.pdf.

References in notes

[57] *Web Services Business Process Execution Language Version 2.0 OASIS Standard*, OASIS, 2007, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf.

[58] M. CARBONE, K. HONDA, N. YOSHIDA. *A Calculus of Global Interaction based on Session Types*, in "Electr. Notes Theor. Comput. Sci.", 2007, vol. 171, n^o 3, p. 127–151.

[59] A. IGARASHI, N. KOBAYASHI. *Resource usage analysis*, in "POPL conference", ACM Press, 2002, p. 331–342.

[60] K. POHL, G. BÖCKLE, F. VAN DER LINDEN. *Software Product Line Engineering: Foundations, Principles, and Techniques Objects*, Springer Verlag, 2005.

[61] D. SANGIORGI. *On the origins of bisimulation and coinduction*, in "ACM Trans. Program. Lang. Syst.", 2009, vol. 31, n^o 4.