# Algorithmics, Programming, Software and Architecture

1. **ABSTRACTION** Project-Team ................................................................. 5
2. **ALGORITHMS** Project-Team ................................................................. 9
3. **AOSTE** Project-Team ........................................................................... 10
4. **CASCADE** Project-Team ....................................................................... 13
5. **CONTRAINTES** Project-Team ............................................................... 15
6. **FORMES** Team ................................................................................... 18
7. **GALLIUM** Project-Team ......................................................................... 21
8. **MOSCOVA** Project-Team ...................................................................... 22
9. **PARKAS** Team ..................................................................................... 25
10. **PLR2** Project-Team ............................................................................... 28
11. **SALSA** Project-Team ........................................................................... 30
12. **SECRET** Project-Team (section vide) ................................................... 31

# Applied Mathematics, Computation and Simulation

13. **CAD** Team .......................................................................................... 32
14. **CLASSIC** Project-Team (section vide) ................................................ 33
15. **GAMMA3** Project-Team (section vide) ............................................... 34
16. **MATHFI** Project-Team ......................................................................... 35
17. **MICMAC** Project-Team (section vide) .................................................. 40
18. **POEMS** Project-Team .......................................................................... 41
19. **SIERRA** Project-Team .......................................................................... 43

# Computational Sciences for Biology, Medicine and the Environment

20. **BANG** Project-Team ............................................................................ 46
21. **CLIME** Project-Team .......................................................................... 47
22. **ESTIME** Project-Team ......................................................................... 49
23. **MACS** Project-Team ............................................................................ 51
24. **NEUROMATHCOMP** Project-Team ........................................................ 53
25. **REO** Project-Team ............................................................................... 56
26. **SISYPHE** Project-Team ......................................................................... 57

# Networks, Systems and Services, Distributed Computing

27. **ARLES** Project-Team .......................................................................... 59
28. **GANG** Project-Team (section vide) ...................................................... 64
29. **HIPECOM** Project-Team ...................................................................... 65
30. **RAP** Project-Team (section vide) .......................................................... 66
31. **REGAL** Project-Team .......................................................................... 67
32. **TREC** Project-Team ............................................................................. 69

# Perception, Cognition, Interaction

33. **ALPAGE** Project-Team ........................................................................ 70
34. **AXIS** Project-Team ............................................................................. 75
35. **IMARA** Project-Team .......................................................................... 79
36. **IMEDIA** Project-Team .......................................................................... 81
37. SMIS Project-Team ................................................................. 82
38. WILLOW Project-Team ............................................................ 83
5. Software

5.1. The Apron Numerical Abstract Domain Library

Participants: Antoine Miné [correspondent], Bertrand Jeannet [team PopArt, INRIA-RA].

The APRON library is dedicated to the static analysis of the numerical variables of a program by abstract interpretation. Its goal is threefold: provide ready-to-use numerical abstractions under a common API for analysis implementers, encourage the research in numerical abstract domains by providing a platform for integration and comparison of domains, and provide a teaching and demonstration tool to disseminate knowledge on abstract interpretation.

The APRON library is not tied to a particular numerical abstraction but instead provides several domains with various precision versus cost trade-offs (including intervals, octagons, linear equalities and polyhedra). A specific C API was designed for domain developers to minimize the effort when incorporating a new abstract domain: only few domain-specific functions need to be implemented while the library provides various generic services and fallback methods (such as scalar and interval operations for most numerical data-types, parametric reduced products, and generic transfer functions for non-linear expressions). For the analysis designer, the APRON library exposes a higher-level API with C, C++, OCaml, and Java bindings. This API is domain-neutral and supports a rich set of semantic operations, including parallel assignments (useful to analyze automata), substitutions (useful for backward analysis), non-linear numerical expressions, and IEEE floating-point arithmetic.

The APRON library is freely available on the web at http://apron.cri.ensmp.fr/library; it is distributed under the LGPL license and is hosted at INRIAGForge. Packages exist for the Debian and Fedora Linux distributions. In order to help disseminate the knowledge on abstract interpretation, a simple inter-procedural static analyzer for a toy language is included. An on-line version is deployed at http://pop-art.inrialpes.fr/interproc/interprocweb.cgi.

The APRON library is developed since 2006 and currently consists of 130 000 lines of C, C++, OCaml, and Java.

Current and past external library users include the Constraint team (LINA, Nantes, France), the Proval/Démon team (LRI Orsay, France), the Analysis of Computer Systems Group (New-York University, USA), the Sierum software analysis platform (Kansas State University, USA), NEC Labs (Princeton, USA), EADS CCR (Paris, France), IRIT (Toulouse, France), ONERA (Toulouse, France), CEA LIST (Saclay, France), VERIMAG (Grenoble, France), ENSMP CRI (Fontainebleau, France), the IBM T.J. Watson Research Center (USA), the University of Edinburgh (UK).

5.2. The Astrée Static Analyzer of Synchronous Software

Participants: Patrick Cousot [project scientifique leader, correspondent], Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, Xavier Rival.

ASTRÉE is a static analyzer for sequential programs based on abstract interpretation [37], [28], [38], [30].

The ASTRÉE static analyzer [27], [43] [1] www.astree.ens.fr aims at proving the absence of runtime errors in programs written in the C programming language.

ASTRÉE analyzes structured C programs, with complex memory usages, but without dynamic memory allocation nor recursion. This encompasses many embedded programs as found in earth transportation, nuclear energy, medical instrumentation, and aerospace applications, in particular synchronous control/command. The whole analysis process is entirely automatic.
**ASTRÉE** discovers all runtime errors including:

- undefined behaviors in the terms of the ANSI C99 norm of the C language (such as division by 0 or out of bounds array indexing);
- any violation of the implementation-specific behavior as defined in the relevant Application Binary Interface (such as the size of integers and arithmetic overflows);
- any potentially harmful or incorrect use of C violating optional user-defined programming guidelines (such as no modular arithmetic for integers, even though this might be the hardware choice);
- failure of user-defined assertions.

The analyzer performs an abstract interpretation of the programs being analyzed, using a parametric domain (**ASTRÉE** is able to choose the right instantiation of the domain for wide families of software). This analysis produces abstract invariants, which over-approximate the reachable states of the program, so that it is possible to derive an over-approximation of the dangerous states (defined as states where any runtime error mentioned above may occur) that the program may reach, and produces alarms for each such possible runtime error. Thus the analysis is sound (it correctly discovers all runtime errors), yet incomplete, that is it may report false alarms (i.e., alarms that correspond to no real program execution). However, the design of the analyzer ensures a high level of precision on domain-specific families of software, which means that the analyzer produces few or no false alarms on such programs.

In order to achieve this high level of precision, **ASTRÉE** uses a large number of expressive abstract domains, which allow expressing and inferring complex properties about the programs being analyzed, such as numerical properties (digital filters, floating-point computations), Boolean control properties, and properties based on the history of program executions.

**ASTRÉE** has achieved the following two unprecedented results:

- **A340–300.** In Nov. 2003, **ASTRÉE** was able to prove completely automatically the absence of any RTE in the primary flight control software of the Airbus A340 fly-by-wire system, a program of 132,000 lines of C analyzed in 1h20 on a 2.8 GHz 32-bit PC using 300 MB of memory (and 50mn on a 64-bit AMD Athlon 64 using 580 MB of memory).
- **A380.** From Jan. 2004 on, **ASTRÉE** was extended to analyze the electric flight control codes then in development and test for the A380 series. The operational application by Airbus France at the end of 2004 was just in time before the A380 maiden flight on Wednesday, 27 April, 2005.

These research and development successes have led to consider the inclusion of **ASTRÉE** in the production of the critical software for the A350. **ASTRÉE** is currently industrialized by AbsInt Angewandte Informatik GmbH and is commercially available.

### 5.3. The AstréeA Static Analyzer of Asynchronous Software

**Participants:** Patrick Cousot [project scientifique leader, correspondant], Radhia Cousot, Jérôme Feret, Antoine Miné, Xavier Rival.

**ASTRÉE A** is a static analyzer prototype for parallel software based on abstract interpretation [39], [40], [32]. It started with support from Thésée ANR project (2006–2010) and is continuing within the ASTRÉE A project (2012–2015).

The **ASTRÉE A** prototype [www.astreea.ens.fr](http://www.astreea.ens.fr) is a fork of the **ASTRÉE** static analyzer (see 5.2) that adds support for analyzing parallel embedded C software.
**ASTRÉEA** analyzes C programs composed of a fixed set of threads that communicate through a shared memory and synchronization primitives (mutexes, FIFOs, blackboards, etc.), but without recursion nor dynamic creation of memory, threads nor synchronization objects. **ASTRÉEA** assumes a real-time scheduler, where thread scheduling strictly obeys the fixed priority of threads. Our model follows the ARINC 653 OS specification used in embedded industrial aeronautic software. Additionally, **ASTRÉEA** employs a weakly-consistent memory semantics to model memory accesses not protected by a mutex, in order to take into account soundly hardware and compiler-level program transformations (such as optimizations). **ASTRÉEA** checks for the same run-time errors as **ASTREE**, with the addition of data-races.

Compared to **ASTREE**, **ASTRÉEA** features: a new iterator to compute thread interactions, a refined memory abstraction that takes into account the effect of interfering threads, and a new scheduler partitioning domain. This last domain allows discovering and exploiting mutual exclusion properties (enforced either explicitly through synchronization primitives, or implicitly by thread priorities) to achieve a precise analysis. 

**ASTRÉEA** is currently being applied to analyze a large industrial avionic software: 1.6 MLines of C and 15 threads, completed with a 2,500-line model of the ARINC 653 OS developed for the analysis. The analysis currently takes 29h on a 2.66 GHz 64-bit intel server using one core and generates around 1,800 alarms. The low computation time (only a few times larger than the analysis time by **ASTREE** of synchronous programs of a similar size and structure) shows the scalability of the approach (in particular, we avoid the usual combinatorial explosion associated to thread interleavings). Precision-wise, the result, while not as impressive as that of **ASTREE**, is quite encouraging. Improvements were made this year concerning the precision of **ASTRÉEA** (from 7,600 alarms in 2010 to 1,800 now) and will continue within the scope of the **ASTRÉEA** ANR project (Section 8.1.1.2).

The details of the analysis are described in [22].

### 5.4. OpenKappa

**Participants:** Monte Brown [Harvard Medical School], Vincent Danos [University of Edinburgh], Jérôme Feret [Correspondent], Walter Fontana [Harvard Medical School], Russ Harmer [Harvard Medical School], Jean Krivine [Paris VII].

**OPENKAPPA** is a collection of tools to build, debug and run models of biological pathways. It contains a compiler for the Kappa Language [49], a static analyzer [48] (for debugging models), a simulator [47], a compression tool for causal traces [46], and a model reduction tool [4], [45], [50].

**OPENKAPPA** is developed since 2007 and, the OCaml version currently consists of 46 000 lines of OCaml. Software are available in OCaml and in Java. Moreover, an Eclipse plug-in is available. **OPENKAPPA** is freely available on the web at [http://kappalanguage.org](http://kappalanguage.org) under the LGPL license. Discussion groups are also available on line.

Current external users include the Ecole Polytechnique Federale de Lausanne, the UNAM-Genomics Mexico team. It is used as pedagogical material in graduate lessons at Harvard Medical School, and at the Interdisciplinary Approaches to Life science (AIV) Master Program (Université de Médecine Paris-Descartes).

### 5.5. Translation Validation

**Participant:** Xavier Rival [correspondent].

The main goal of this software project is to make it possible to certify automatically the compilation of large safety critical software, by proving that the compiled code is correct with respect to the source code: When the proof succeeds, this guarantees that no compiler bug did cause incorrect code be generated. Furthermore, this approach should allow to meet some domain specific software qualification criteria (such as those in DO-178 regulations for avionics software), since it allows proving that successive development levels are correct with respect to each other i.e., that they implement the same specification. Last, this technique also justifies the use of source level static analyses, even when an assembly level certification would be required, since it establishes separately that the source and the compiled code are equivalent.
The compilation certification process is performed automatically, thanks to a prover designed specifically. The automatic proof is done at a level of abstraction which has been defined so that the result of the proof of equivalence is strong enough for the goals mentioned above and so that the proof obligations can be solved by efficient algorithms.

The current software features both a C to Power-PC compilation certifier and an interface for an alternate source language frontend, which can be provided by an end-user.

5.6. Zarith

Participants: Antoine Miné [Correspondent], Xavier Leroy [INRIA Paris-Rocquencourt], Pascal Cuoq [CEA LIST].

ZARITH is a small (10K lines) OCaml library that implements arithmetic and logical operations over arbitrary-precision integers. It is based on the GNU MP library to efficiently implement arithmetic over big integers. Special care has been taken to ensure the efficiency of the library also for small integers: small integers are represented as Caml unboxed integers and use a specific C code path. Moreover, optimized assembly versions of small integer operations are provided for a few common architectures.

ZARITH is an open-source project hosted at OCamlForge (http://forge.ocamlcore.org/projects/zarith) and distributed under a modified LGPL license.

ZARITH is currently used in the ASTRÉE analyzer to enable the sound analysis of programs featuring 64-bit (or larger) integers. It is also used in the Frama-C analyzer platform developed at CEA LIST and INRIA Saclay.
4. Software

4.1. Algolib

The Algolib library is a set of Maple routines that have been developed in the project for more than 15 years. Several parts of it have been incorporated into the standard library of Maple, but the most up-to-date version is always available for free from our web pages http://algo.inria.fr/libraries/. This library provides: tools for combinatorial structures (the combstruct package), including enumeration, random or exhaustive generation, generating functions for a large class of attribute grammars; tools for linear difference and differential equations (the gfun package), which have received a very positive review in Computing Reviews and have been incorporated in N. Sloane's superseeker at Bell Labs; tools for systems of multivariate linear operators, definite sums and integrals (the Mgfun package), including Gröbner bases in Ore algebras, that also treat commutative polynomials and have been the standard way to solve polynomial systems in Maple for a long period (although the user would not notice it); Mgfun has also been chosen at Risc (Linz) as the basis for their package Dasing, tools for expansions in general asymptotic scales, which make it possible to handle in a transparent and automatic way the problems of finding the proper scale for an expansion and of dealing with the indefinite cancellation problem (the MultiSeries package).

4.2. Mathematics on the Web

We also provide access to our work to scientists who are not using Maple or any other computer algebra system in the form of automatically generated encyclopedias available on the web. The Encyclopedia of Combinatorial Structures at http://algo.inria.fr/ecs/ thus contains more than 1000 combinatorial structures for which generating functions, enumeration sequences, recurrences, and asymptotic approximations have been computed automatically. It gets more than 16,000 hits per month. The Dynamic Dictionary of Mathematical Functions (DDMF) at http://ddmf.msr-inria.inria.fr/ gathers several dozens of special functions for which identities, guaranteed high-precision numerical evaluations, power-series and asymptotic expansions, graphs, ... are generated automatically and on the user’s request, starting from a linear differential equation and its initial conditions. The underlying symbolic algorithms and implementations are those of gfun and Mgfun. All the production process being automated, the difficult and expensive step of checking each formula individually is suppressed. A nice specificity of this encyclopedia is its interactivity: the approximations values (numbers, series) are not bound to a statically set precision, rather, the user can fill in the precision he wants in a form, before clicking to ask for a refined identity to be generated, then displayed. This interactivity is based on a tool DynaMoW at http://ddmf.msr-inria.inria.fr/DynaMoW/ (for Dynamic Mathematics on the Web) that we develop as well. This is an Ocaml library that simultaneously controls external symbolic calculations and webpage generation at the same time. Being available on the web, the DDMF also plays the role of a showcase for part of the packages developed in our project. It is a successor of our former Encyclopedia of Special Functions at http://algo.inria.fr/esf/.
5. Software

5.1. TimeSquare

Participants: Charles André, Nicolas Chleq, Julien Deantoni, Benoît Ferrero, Frédéric Mallet [correspondant].

TimeSquare is a software environment for the modeling and analysis of timing constraints in embedded systems. It relies specifically on the Time Model of the MARTE UML profile (see section 3.2), and more accurately on the associated Clock Constraint Specification Language (CCSL) for the expression of timing constraints.

TimeSquare offers four main functionalities:

1. graphical and/or textual interactive specification of logical clocks and relative constraints between them;
2. definition and handling of user-defined clock constraint libraries;
3. automated simulation of concurrent behavior traces respecting such constraints, using a Boolean solver for consistent trace extraction;
4. call-back mechanisms for the traceability of results (animation of models, display and interaction with waveform representations, generation of sequence diagrams...).

In practice TimeSquare is a plug-in developed with Eclipse modeling tools. The software is registered by the Agence pour la Protection des Programmes, under number IDDN.FR.001.170007.000.S.P.2009.001.10600. It can be downloaded from the site http://timesquare.inria.fr/. It has been integrated in the OpenEmbeDD ANR RNTL platform, and other such actions are under way.

5.2. K-Passa

Participants: Anthony Coadou, Jean-Vivien Millo [correspondant], Robert de Simone.

This software is dedicated to the simulation, analysis, and static scheduling scheduling of Event/Marked Graphs, SDF and KRG extensions. A graphical interface allows to edit the Process Networks and their time annotations (latency, ...). Symbolic simulation and graph-theoretic analysis methods allow to compute and optimize static schedules, with best throughputs and minimal buffer sizes. In the case of KRG the (ultimately k-periodic) routing patterns can also be provided and transformed for optimal combination of switching and scheduling when channels are shared. KPASSA also allows for import/export of specific description formats such as UML-MARTE, to and from our other TimeSquare tool.

The tool was originally developed mainly as support for experimentations following our research results on the topic of Latency-Insensitive Design. This research was conducted and funded in part in the context of the CIM PACA initiative, with initial support from ST Microelectronics and Texas Instruments.

KPASSA is registered by the Agence pour la Protection des Programmes, under the number IDDN.FR.001.310003.000.S.P.2009.000.20700. it can be downloaded from the site http://www-sop.inria.fr/aoste/index.php?page=software/kpassa.

5.3. SynDEx

Participants: Maxence Guesdon, Yves Sorel [correspondant], Cécile Stentzel, Meriem Zidouni.

SynDEx is a system level CAD software implementing the AAA methodology for rapid prototyping and for optimizing distributed real-time embedded applications. Developed in OCaML it can be downloaded free of charge, under INRIA copyright, from the general SynDEx site http://www.syndex.org.
The AAA methodology is described in section 3.3. Accordingly, SYNDEX explores the space of possible allocations (spatial distribution and temporal scheduling), from application elements to architecture resources and services, in order to match real-time requirements; it does so by using schedulability analyses and heuristic techniques. Ultimately it generates automatically distributed real-time code running on real embedded platforms. The last major release of SYNDEX (V7) allows the specification of multi-periodic applications.

Application algorithms can be edited graphically as directed acyclic task graphs (DAG) where each edge represent a data dependence between tasks, or they may be obtained by translations from several formalisms such as Scicos (http://www.scicos.org), Signal/Polychrony (http://www.irisa.fr/espresso/Polychrony), or UML2/MARTE models (http://www.omg.org/technology/documents/profile_catalog.htm). Architectures are represented as graphical block diagrams composed of programmable (processors) and non-programmable (ASIC, FPGA) computing components, interconnected by communication media (shared memories, links and busses for message passing). In order to deal with heterogeneous architectures it may feature several components of the same kind but with different characteristics.

Two types of non-functional properties can be specified for each task of the algorithm graph. First, a period that does not depend on the hardware architecture. Second, real-time features that depend on the different types of hardware components, ranging amongst execution and data transfer time, memory, etc. Requirements are generally constraints on deadline equal to period, latency between any pair of tasks in the algorithm graph, dependence between tasks, etc.

Exploration of alternative allocations of the algorithm onto the architecture may be performed manually and/or automatically. The latter is achieved by performing real-time multiprocessor schedulability analyses and optimization heuristics based on the minimization of temporal or resource criteria. For example while satisfying deadlines and latencies constraints they can minimize the total execution time (makespan) of the application onto the given architecture, as well as the amount of memory. The results of each exploration is visualized as timing diagrams simulating the distributed real-time implementation.

Finally, real-time distributed embedded code can be automatically generated for dedicated distributed real-time executives, possibly calling services of resident real-time operating systems such as Linux/RTAI or Osek for instance. These executives are deadlock-free, based on off-line scheduling policies. Dedicated executives induce minimal overhead, and are built from processor-dependent executive kernels. To this date, executives kernels are provided for: TMS320C40, PIC18F2680, i80386, MC68332, MPC555, i80C196 and Unix/Linux workstations. Executive kernels for other processors can be achieved at reasonable cost following these examples as patterns.

5.4. SAS

**Participants:** Daniel de Rauglaudre [correspondant], Yves Sorel.

The SAS (Simulation and Analysis of Scheduling) software allows the user to perform the schedulability analysis of periodic task systems in the monoprocessor case.

The main contribution of SAS, when compared to other commercial and academic softwares of the same kind, is that it takes into account the exact preemption cost between tasks during the schedulability analysis. Beside usual real-time constraints (precedence, strict periodicity, latency, etc.) and fixed-priority scheduling policies (Rate Monotonic, Deadline Monotonic, Audsley++ User priorities), SAS additionally allows to select dynamic scheduling policy algorithms such as Earliest Deadline First (EDF). The resulting schedule is displayed as a typical Gantt chart with a transient and a permanent phase, or as a disk shape called "dameid", which clearly highlights the idle slots of the processor in the permanent phase.

For a schedulable task system under EDF, when the exact preemption cost is considered, the period of the permanent phase may be much longer than the least common multiple (LCM) of the periods of all tasks, as often found in traditional scheduling theory. Specific effort has been made to improve display in this case. The classical utilization factor, the permanent exact utilization factor, the preemption cost in the permanent phase, and the worst response time for each task are all displayed when the system is schedulable. Response times of each task relative time can also be displayed (separately).
SAS is written in OCaML, using CAMLP5 (syntactic preprocessor) and OLIBRT (a graphic toolkit under X). Both are written by Daniel de Rauglaudre.
5. Software

5.1. MitMTool

Participants: Charles Bouillaguet, Patrick Derbez, Pierre-Alain Fouque.

The purpose of MitMTool is to look for guess-and-determine and meet-in-the-middle attacks on AES and AES-based constructions. This tool allows us to improve known attacks on round-reduced versions of AES, on the LEX stream-cipher on the PELICAN Message Authentication Code and on fault attack on AES. Basically, it solves the problem to find all the solutions of a linear system of equations on the variables \( x \) and \( S(x) \) where \( S \) is an invert function. The tool allows to compute the complexity of some good attack as well as the C code of the attack. We verify that the complexity estimates are accurate using experiments. We also use it to find one solution of the system for chosen-key differential attacks. There are mainly two tools: the first one only looks for guess-and-determine attack and tries to propagate some knowledge and guesses value when it cannot find automatically the value of some variable. The second tool uses the technique of the first tool and more advanced technique to take into account attacks with memory that use the meet-in-the-middle attack.

5.2. ProVerif

Participants: Bruno Blanchet, Vincent Cheval.

ProVerif (www.proverif.ens.fr) is an automatic security protocol verifier, in the formal model (so called Dolev–Yao model). In this model, cryptographic primitives are considered as black boxes. This protocol verifier is based on an abstract representation of the protocol by Horn clauses. Its main features are:

- It can handle many different cryptographic primitives, including shared- and public-key cryptography (encryption and signatures), hash functions, and Diffie–Hellman key agreements, specified both as rewrite rules or as equations.
- It can handle an unbounded number of sessions of the protocol (even in parallel) and an unbounded message space. This result has been obtained thanks to some well-chosen approximations. This means the verifier can give false attacks, but if it claims that the protocol satisfies some property, then the property is actually satisfied. ProVerif also provides attack reconstruction: when it cannot prove a property, it tries to reconstruct an attack, that is, an execution trace of the protocol that falsifies the desired property.

The ProVerif verifier can prove the following properties:

- secrecy (the adversary cannot obtain the secret);
- authentication and more generally correspondence properties, of the form "if an event has been executed, then other events have been executed as well";
- strong secrecy (the adversary does not see the difference when the value of the secret changes);
- equivalences between processes that differ only by terms;

ProVerif has been used by researchers for studying various kinds of protocols, including electronic voting protocols, certified email protocols, and zero-knowledge protocols. It has been used as a back-end for the tool Tulafale implemented at Microsoft Research Cambridge, which verifies web services protocols. It has also been used as a back-end for verifying implementations of protocols in F# (a dialect of ML included in .NET), by Microsoft Research Cambridge and the joint INRIA-Microsoft research center.

ProVerif is freely available on the web, at www.proverif.ens.fr, under the GPL license.
5.3. CryptoVerif

Participants: Bruno Blanchet, David Cadé.

CryptoVerif (www.cryptoverif.ens.fr) is an automatic protocol prover sound in the computational model. In this model, messages are bitstrings and the adversary is a polynomial-time probabilistic Turing machine. CryptoVerif can prove:
- secrecy;
- correspondences, which include in particular authentication.

CryptoVerif provides a generic mechanism for specifying the security assumptions on cryptographic primitives, which can handle in particular symmetric encryption, message authentication codes, public-key encryption, signatures, hash functions, Diffie-Hellman key agreement.

The generated proofs are proofs by sequences of games, as used by cryptographers. These proofs are valid for a number of sessions polynomial in the security parameter, in the presence of an active adversary. CryptoVerif can also evaluate the probability of success of an attack against the protocol as a function of the probability of breaking each cryptographic primitive and of the number of sessions (exact security).

CryptoVerif is still at a rather early stage of development, but it has already been used for a study of Kerberos in the computational model. It is also used as a back-end for verifying implementations of protocols in F# at Microsoft Research Cambridge and at the joint INRIA-Microsoft research center.

CryptoVerif is freely available on the web, at www.cryptoverif.ens.fr, under the CeCILL-B license.
5. Software

5.1. BIOCHAM

Participants: François Fages, Steven Gay, Dragana Jovanovska, Aurélien Rizk, Sylvain Soliman.

The Biochemical Abstract Machine BIOCHAM [29] is a modeling environment for systems biology distributed as open-source since 2003. Current version is v3.3. BIOCHAM uses a compositional rule-based language for modeling biochemical systems, allowing patterns for expressing set of rules in a compact form. This rule-based language is compatible with the Systems Biology Markup Language (SBML) and is interpreted with three semantics corresponding to three abstraction levels:

1. the boolean semantics (presence or absence of molecules),
2. the differential semantics (concentrations of molecules),
3. the stochastic semantics (discrete numbers of molecules).

Based on this formal framework, BIOCHAM features:

- Boolean and numerical simulators (Rosenbrock’s method for the differential semantics, Gillespie’s algorithm with tau lipping for the stochastic semantics);
- a temporal logic language (CTL for qualitative models and QFLTL(R) with numerical constraints for quantitative models) for formalizing biological properties such as reachability, checkpoints, oscillations or stability, and checking them automatically with model-checking techniques;
- automatic search procedures to infer parameter values, initial conditions and even reaction rules from temporal logic properties;
- automatic detection of invariants, through constraint-based analysis of the underlying Petri net;
- an SBGN-compatible reaction graph editor;
- an event handler allowing the encoding of hybrid models and formalisms [30].

BIOCHAM is implemented in GNU-Prolog and interfaced to the symbolic model checker NuSMV and to the continuous optimization tool CMAES developed by the EPI TAO.

5.2. Nicotine

Participant: Sylvain Soliman.

Nicotine is a GNU Prolog framework dedicated to the analysis of Petri nets. It was originally built for the computation of invariants using GNU Prolog’s CLP(FD) solver but has been further extended to allow import/export of various Petri nets formats. It provides as independent modules different features that can sometimes also be integrated in BIOCHAM, like SEPI computation, or left aside, like unambiguous ODE to Petri net conversion, since a more general heuristic conversion is developed for BIOCHAM.

5.3. Spatio-temporal simulation environment (STSE)

Participant: Szymon Stoma.

The overall goal of this project is to provide a software platform gathering a set of open-source tools and workflows facilitating spatio-temporal simulations (preferably of biological systems) based on microscopy data. The framework currently contains modules to digitize, represent, analyze, and model spatial distributions of molecules in static and dynamic structures (e.g. growing). A strong accent is put on the experimental verification of biological models by actual, spatio-temporal data acquired using microscopy techniques. Project was initially started at Humboldt University Berlin and moved to INRIA with its founder. Project webpage is: http://stse-software.org.
5.4. YeastTracker

**Participant:** Jannis Uhlendorf.

YeastTracker is a software to follow single cells in movies and to quantify fluorescent images based on this tracking. It has been developed for yeast cells, but is also applicable to other cells that have a defined round shape. The software is written in Matlab and uses a circular Hough transform and binary integer programming to detect and follow cells. It allows to quantify the mean fluorescence of each cell as well as the co-localization of two different fluorescent markers. The software is available on request (jannis.uhlendorf@inria.fr).

5.5. Rules2CP

**Participants:** François Fages, Raphaël Martin.

Rules2CP is a rule-based modeling language for constraint programming. It is distributed since 2009 as open-source. Unlike other modeling languages for constraint programming, Rules2CP adopts a single knowledge representation paradigm based on rules without recursion, and a restricted set of data structures based on records and enumerated lists given with iterators. This allows us to model complex constraint satisfaction problems together with search strategies, where search trees are expressed by logical formulae and heuristic choice criteria are defined with preference orderings by pattern-matching on the rules’ left-hand sides.

The expressiveness of Rules2CP has been illustrated in the FP6 Strep project **Net-WMS** by a complete library for packing problems, called PKML (Packing Knowledge Modeling Library), which, in addition to pure bin packing and bin design problems, can deal with common sense rules about weights, stability, as well as specific packing business rules.

5.6. SiLCC

**Participant:** Thierry Martinez.

SiLCC is an extensible modular concurrent constraint programming language relying upon linear logic. It is a complete implementation of the Linear logic Concurrent Constraint programming paradigm of Saraswat and Lincoln using the formal semantics of Fages, Ruet and Soliman. It is a single-paradigm logical language, enjoying concurrency, imperative traits, and a clean module system allowing to develop hierarchies of constraint systems within the language.

This software prototype is used to study the design of hierarchies of extensible libraries of constraint solvers. SiLCC is also considered as a possible implementation language for restructuring the code of **BIOCHAM**.

5.7. EMoP

**Participant:** Thierry Martinez.

EMoP is an extension of Prolog with first-class modules. These modules have the formal semantics of the LCC modules and provide Prolog with notions of namespaces, closures and objects within a simple programming model. Modules are also the support for user-definition of macros and modular syntax extensions. EMoP is bootstrapped and uses the GNU Prolog compilation chain as back-end.

5.8. CHRat

**Participant:** Thierry Martinez.

CHRat is a modular version of the well known Constraint Handling Rules language CHR, called for CHRat for CHR with *ask* and *tell*. Inspired by the LCC framework, this extension of CHR makes it possible to reuse CHRat components both in rules and guards in other CHRat components, and define hierarchies of constraint solvers. CHRat is a bootstrapped preprocessor for CHR which generates code for SWI/Prolog.
5.9. CLPGUI

**Participant:** François Fages.

CLPGUI is a generic graphical user interface written in Java for constraint logic programming. It is available for GNU-Prolog and SICStus Prolog. CLPGUI has been developed both for teaching purposes and for debugging complex programs. The graphical user interface is composed of several windows: one main console and several dynamic 2D and 3D viewers of the search tree and of finite domain variables. With CLPGUI it is possible to execute incrementally any goal, backtrack or recompute any state represented as a node in the search tree. The level of granularity for displaying the search tree is defined by annotations in the CLP program.

CLPGUI has been mainly developed in 2001 and is distributed as third-party software on GNU-Prolog and SICStus Prolog web sites. In 2009, CLPGUI has been interfaced to Rules2CP/PKML and used in the FP6 Strep Net-WMS with a non-released version.
5. Software

5.1. aCiNO

Participants: Fei He [correspondant], Min Zhou.

aCiNO is an SMT (Satisfiability Modulo Theory) solver based on a Nelson-Oppen [65] architecture, and written in C++. Currently, two popular theories are considered: linear real arithmetic (LRA) and uninterpreted functions (UF). A lazy approach is used for solving SMT problem. For efficiency consideration, the solver is implemented in an incremental way. It also invokes an online SAT solver, which is now a modified MiniSAT, so that recovery from conflict is possible.

5.2. CoLoR and Rainbow

Participants: Frédéric Blanqui [correspondant], Kim-Quyen Ly, Sidi Ould Biha.

CoLoR is a Coq [44] library on rewriting theory and termination of nearly 70,000 lines of code [11]. It provides definitions and theorems for:

- Mathematical structures: relations, (ordered) semi-rings.
- Data structures: lists, vectors, polynomials with multiple variables, finite multisets, matrices.
- Term structures: strings, algebraic terms with symbols of fixed arity, algebraic terms with varyadic symbols, simply typed lambda-terms.
- Transformation techniques: conversion from strings to algebraic terms, conversion from algebraic to varyadic terms, arguments filtering, rule elimination, dependency pairs, dependency graph decomposition, semantic labelling.
- Termination criteria: polynomial interpretations, multiset ordering, lexicographic ordering, first and higher order recursive path ordering, matrix interpretations.

Rainbow is a tool for automatically certifying termination certificates expressed in the CPF XML format [29] used in the termination competition on termination [32]. Termination certificates are translated and checked in Coq by using the CoLoR library.

CoLoR and Rainbow are distributed under the CeCILL license on http://color.inria.fr/ . Various people participated to its development (see the website for more information).

5.3. EDOLA

Participants: Hehua Zhang [correspondant], Ming Gu, Hui Kong, Yu Jiang.

Joint work with Jiaguang Sun (Tsinghua University, China).

EDOLA [26] is an integrated tool for domain-specific modeling and verification of PLC applications [74]. It is based on a domain-specific modeling language to describe system models. It supports both model checking and automatic theorem proving techniques for verification. The goal of this tool is to possess both the usability in domain modeling, the reusability in its architecture and the capability of automatic verification.

For the moment, we have developed a prototype of the EDOLA language, which can easily describe the features of PLC applications like the scan cycle mechanism, the pattern of environment model, time constraints and five property patterns. TLA+ [59] was chosen as the intermediate language to implement the automatic verification of EDOLA models. A prototype of EDOLA has also been developed, which comes along with an editor to help writing EDOLA models. To automatically verify properties on EDOLA models, it provides the interface for both a model checker TLC [59] and a first-order theorem prover SPASS [75].
5.4. Moca

Participant: Frédéric Blanqui [correspondant].

Joint work with Pierre Weis (INRIA Rocquencourt) and Richard Bonichon (CEA).

Moca is a construction functions generator for OCaml [60] data types with invariants. It allows the high-level definition and automatic management of complex invariants for data types. In addition, it provides the automatic generation of maximally shared values, independently or in conjunction with the declared invariants.

A relational data type is a concrete data type that declares invariants or relations that are verified by its constructors. For each relational data type definition, Moca compiles a set of construction functions that implements the declared relations.

Moca supports two kinds of relations:
- predefined algebraic relations (such as associativity or commutativity of a binary constructor),
- user-defined rewrite rules that map some pattern of constructors and variables to some arbitrary user’s define expression.

The properties that user-defined rules should satisfy (completeness, termination, and confluence of the resulting term rewriting system) must be verified by a programmer’s proof before compilation. For the predefined relations, Moca generates construction functions that allow each equivalence class to be uniquely represented by their canonical value.

Moca is distributed under QPL on http://moca.inria.fr/.

5.5. SimSoC

Participant: Vania Joloboff [correspondant].

SimSoC is an infrastructure to run simulation models which comes along with a library of simulation models. SimSoC allows its users to experiment various system architectures, study hardware/software partition, and develop embedded software in a co-design environment before the hardware is ready to be used. SimSoC aims at providing high performance, yet accurate simulation, and provide tools to evaluate performance and functional or non functional properties of the simulated system.

SimSoC is based on SystemC standard and uses Transaction Level Modeling for interactions between the simulation models. The current version of SimSoC is based on the open source libraries from the OSCI Consortium: SystemC version 2.2 and TLM 2.0.1 [54], [33]. Hardware components are modeled as TLM models, and since TLM is itself based on SystemC, the simulation is driven by the SystemC kernel. We use standard, unmodified, SystemC (version 2.2), hence the simulator has a single simulation loop.

The second open source version of SimSoC, SimSoC v0.7.1, has been released in November 2010. It contains a full simulator for ARM V5 and PowerPC both running at an average speed of about 80 Millions instructions per second in, and a simulator for the MIPS architecture with an average speed of 20 Mips in mode DT1. It represents about 70,000 lines of source code and includes:
- Instruction Set Simulators. The ARM Version 5 architecture has been implemented with DT0, DT1, DT2 mode. The ARM and PowerPC 600 architecture with DT0 and DT1 mode. For both architectures, complete simulation models of the processor and MMU are provided, making it possible to run operating systems of the simulated platform. MIPS architecture in DT0 mode is under development.
- A dynamic translator from binary programs to an internal representation. For the ARM architecture a compiler has been developed that generates the C++ translated code (for DT2), using parametrized specialization options.
Peripheral models including a serial line controller, a flash memory controller, an interrupt controller.

A utility to generate permanent storage for flash memory simulation; a compiler tool to generate instruction binary decoder.

Examples illustrating the use of the library and infrastructure.

SimSoC is distributed under LGPL on https://gforge.inria.fr/projects/simsoc.

5.6. SimSoC-Cert

Participants: Frédéric Blanqui, Vania Joloboff, Jean-François Monin [correspondant], Xiaomu Shi.

SimSoC-Cert is a set of tools that can automatically generate in various target languages (Coq and C) the decoding functions and the state transition functions of each instruction and addressing mode of the ARMv6 architecture manual [28] (implemented by the ARM11 processor family) but the Thumb and coprocessor instructions. The input of SimSoC-Cert is the ARMv6 architecture manual itself.

Based on this, we first developed simlight (8000 generated lines of C, plus 1500 hand-written lines of C), a simulator for ARMv6 programs using no peripheral and no coprocessor. Next, we developed simlight2, a fast ARMv6 simulator integrated inside a SystemC/TLM module, now part of SimSoC v0.7.

We can also generate similar programs for SH4 [31] but this is still under test.
5. Software

5.1. OCaml

Participants: Xavier Leroy [correspondant], Xavier Clerc [team SED], Damien Doligez, Alain Frisch [LexiFi], Jacques Garrigue [Nagoya University], Maxence Guessedon [team SED], Luc Maranget [EPI Moscova], Michel Mauny [ENSTA], Nicolas Pouillard, Pierre Weis [EPI Estime].

OCaml, formerly known as Objective Caml, is our flagship implementation of the Caml language. From a language standpoint, it extends the core Caml language with a fully-fledged object and class layer, as well as a powerful module system, all joined together by a sound, polymorphic type system featuring type inference. The OCaml system is an industrial-strength implementation of this language, featuring a high-performance native-code compiler for several processor architectures (IA32, AMD64, PowerPC, ARM, etc) as well as a bytecode compiler and interactive loop for quick development and portability. The OCaml distribution includes a standard library and a number of programming tools: replay debugger, lexer and parser generators, documentation generator, compilation manager, and the Camlp4 source pre-processor.


5.2. CompCert C

Participants: Xavier Leroy [correspondant], Sandrine Blazy [EPI Celtique], Alexandre Pilkiewicz.

The CompCert C verified compiler is a compiler for a large subset of the C programming language that generates code for the PowerPC, ARM and x86 processors. The distinguishing feature of CompCert is that it has been formally verified using the Coq proof assistant: the generated assembly code is formally guaranteed to behave as prescribed by the semantics of the source C code. The subset of C supported is quite large, including all C types except long long and long double, all C operators, almost all control structures (the only exception is unstructured switch), and the full power of functions (including function pointers and recursive functions but not variadic functions). The generated PowerPC code runs 2–3 times faster than that generated by GCC without optimizations, and only 7% (resp. 12%) slower than GCC at optimization level 1 (resp. 2).


5.3. Zenon

Participant: Damien Doligez.

Zenon is an automatic theorem prover based on the tableaux method. Given a first-order statement as input, it outputs a fully formal proof in the form of a Coq proof script. It has special rules for efficient handling of equality and arbitrary transitive relations. Although still in the prototype stage, it already gives satisfying results on standard automatic-proving benchmarks.

Zenon is designed to be easy to interface with front-end tools (for example integration in an interactive proof assistant), and also to be easily retargetted to output scripts for different frameworks (for example, Isabelle).


5.4. Menhir

Participants: François Pottier [correspondant], Yann Régis-Gianas [U. Paris Diderot].

Menhir is a new LR(1) parser generator for Objective Caml. Menhir improves on its predecessor, ocamlyacc, in many ways: more expressive language of grammars, including EBNF syntax and the ability to parameterize a non-terminal by other symbols; support for full LR(1) parsing, not just LALR(1); ability to explain conflicts in terms of the grammar.

Web site: http://gallium.inria.fr/~fpottier/menhir/.
5. Software

5.1. F7: Refinement Types for F#

**Participants:** Karthikeyan Bhargavan [correspondant], Cédric Fournet [MSR Cambridge], Andrew D. Gordon [MSR Cambridge].

F7 is an enhanced typechecker for the F# programming language that enables static checking of properties expressed as refinement types.

A refinement type is a base type qualified with a logical formula; the formula can express invariants, preconditions, and postconditions. F7 relies on type annotations, including refinements, provided in specific interface files. While checking code, F7 generates many logical problems which it solves by submitting to Z3, an external theorem prover for first-order logic (de Moura and Bjørner 2008). Finally, F7 erases all refinements and yields ordinary F# modules and interfaces.

Our main aim is to use F7 for the verification of security-critical programs. We have used it to verify implementations of access control mechanisms, multi-party secure sessions, cryptographic protocols for web services security and federated authentication, and secure audit logs.

A first version of F7 was released in 2008. In 2011, we revised the F7 libraries and typechecker and ported it to the released version of F# for .NET 4.0. The second version of F7 was released in December 2011.

The typechecker is written in 16000 lines of F#, with an additional cryptographic library of 9000 lines, and sample code of more than 12000 lines.

5.2. JSTY: Logical Auditing of JavaScript Programs

**Participants:** Karthikeyan Bhargavan [correspondant], Sergio Maffeis [Imperial College], Ravinder Shankesi [U. of Illinois at Urbana Champain].

JSTY is a runtime monitoring and logical auditing framework for JavaScript web applications. It has three components: (1) a contract language for JavaScript that enables programmers to annotate their scripts with assumptions and goals written as first-order logic pre- and post-conditions; (2) a runtime monitor implemented as a browser extension in the web browser Chrome that interprets these contracts at runtime and generates proof obligations for an SMT solver; (3) a logical auditor that checks proof obligations and maps counterexamples to violations of program correctness goals.

The target applications for JSTY include browser extensions as well as website scripts. In the case of browser extensions, our goal is to help extension writers to test their code by annotating it with logical contracts and auditing the code with JSTY. For website scripts, our goal is to check whether a website script obeys a generic security policy expressed as pre-conditions on functions in the browser or DOM API. We have used JSTY to analyze a variety of security-critical browser extensions and website scripts and found several vulnerabilities. We are currently incorporating static checking into JSTY.

JSTY is written in about 1000 lines of JavaScript and we plan a public release in 2012.

5.3. OTT: Tool support for the working semanticist

**Participants:** Peter Sewell [U. of Cambridge], Francesco Zappa Nardelli [correspondant].
Ott is a tool for writing definitions of programming languages and calculi. It takes as input a definition of a language syntax and semantics, in a concise and readable ASCII notation that is close to what one would write in informal mathematics. It generates output:

1. a LaTeX source file that defines commands to build a typeset version of the definition;
2. a Coq version of the definition;
3. an Isabelle version of the definition; and
4. a HOL version of the definition.

Additionally, it can be run as a filter, taking a LaTeX/Coq/Isabelle/HOL source file with embedded (symbolic) terms of the defined language, parsing them and replacing them by typeset terms.

The main goal of the Ott tool is to support work on large programming language definitions, where the scale makes it hard to keep a definition internally consistent, and to keep a tight correspondence between a definition and implementations. We also wish to ease rapid prototyping work with smaller calculi, and to make it easier to exchange definitions and definition fragments between groups. The theorem-prover backends should enable a smooth transition between use of informal and formal mathematics.

In collaboration with Peter Sewell (Cambridge University).

The current version of Ott is about 30000 lines of OCaml. The tool is available from http://moscova.inria.fr/~zappa/software/ott (BSD licence).

Since its release in December 2007, the tool has been used in several projects, including a large proof of type preservation for the OCaml language (without modules) done by Scott Owens.

In 2011, apart from minor bug-fixes and features added, we implemented several performance improvements which result in a up-to 6x speed-up, and kept the Isabelle and Coq backend up-to date with the theorem prover evolution.

The currently released version is 0.21.1.

### 5.4. Lem, a tool for lightweight executable mathematics

**Participants:** Scott Owens [U. of Cambridge], Peter Sewell [U. of Cambridge], Francesco Zappa Nardelli [correspondent].

Lem is a lightweight tool for writing, managing, and publishing large scale semantic definitions. It is also intended as an intermediate language for generating definitions from domain-specific tools, and for porting definitions between interactive theorem proving systems (such as Coq, HOL4, and Isabelle). As such it is a complementary tool to Ott.

Lem resembles a pure subset of Objective Caml, supporting typical functional programming constructs, including top-level parametric polymorphism, datatypes, records, higher-order functions, and pattern matching. It also supports common logical mechanisms including list and set comprehensions, universal and existential quantifiers, and inductively defined relations. From this, Lem generates OCaml, HOL4 and Isabelle code; the OCaml backend uses a finite set library (and does not yet support inductive relations). A Coq backend is in development.

Lem is already in use at Cambridge and INRIA for research on relaxed-memory concurrency. We are currently preparing a feature-complete release with back-ends for HOL4, Isabelle/HOL, Coq, OCaml, and LaTeX. The project web-page is http://www.cl.cam.ac.uk/~so294/lem/ . A paper on a Lem prototype appeared in ITP 2011, in the “rough diamond” category [25].

### 5.5. Memevents-Litmus-Diy-Dont

**Participants:** Jade Alglave, Luc Maranget [correspondent], Susmit Sarkar [U. of Cambridge, UK], Peter Sewell [U. of Cambridge, UK].
Luc Maranget is the main developer of the tools suite of project “Weak Memory Models” (cf. the relevant section).

This suite features three subtools *memevents* (model checker), *litmus* (runs tests on actual machines) and *diy* (generate tests from concise specifications). This year saw a new tool and one official releases (with documentation) [33] — see also http://diy.inria.fr. The releases feature all tools except *memevents*, which we wish to keep for ourselves.

This year main extensions are the handling of the ARM architecture and more collaboration between tools. For the latter, the test generator *diy* enrichs tests with meta-data that are exploited by *litmus*, so as to perform binding of test threads to machine processors, intelligent prefetch of data etc. We plan a new release of the tool suite early next year.

A new, independent, “proof of concept” tool, *offence* was written by J. Alglave and L. Maranget, as a support of our publication [30].

This software is available at http://diy.inria.fr/offence.

5.6. Jocaml

**Participants:** Luc Maranget, Xavier Clerc [correspondant].

Jocaml is an implementation of the join-calculus integrated into Ocaml. With respect to previous join-language prototypes, the most salient feature of the new prototype is a better integration into Ocaml. We achieve binary compatibility with Ocaml, moreover Jocaml releases now follow Ocaml releases. See previous year reports for details on Jocaml. The current version is 3.12.1 (released in September [34]) is available at http://jocaml.inria.fr/.

This new release features an extended Jocaml specific library that provide programmers with an easier access to concurrency and distribution:

1. Some utilities to parse command line, organize client-server connection, etc. This code was written partly by Xavier Clerc, engineer at INRIA SED department.

2. Some new abstractions of text channels help for writing text oriented applications.

5.7. Hevea

**Participant:** Luc Maranget [correspondant].

Hevea is a fast translator from full LaTeX to HTML, written in Ocaml. Hevea is highly configurable with commands written in LaTeX. Mathematics are rendered with UNICODE characters for symbols and HTML tables for formatting. Hevea produces HTML 4.0, enriched by css files. Hevea comes with Hacha companion, which produces a set of HTML pages (for instance, one page per chapter). Since it is very efficient and configurable, Hevea is adequate for on-line manuals or teaching courses.

This year saw a few developments around Hevea, mostly for maintenance. Hevea is available at http://hevea.inria.fr/.
PARKAS Team

5. Software

5.1. Lucid Synchrone

Participant: Marc Pouzet [contact].

Lucid Synchrone is a language for the implementation of reactive systems. It is based on the synchronous model of time as provided by Lustre combined with features from ML languages. It provides powerful extensions such as type and clock inference, type-based causality and initialization analysis and allows to arbitrarily mix data-flow systems and hierarchical automata or flows and valued signals.

It is distributed under binary form, at URL http://www.di.ens.fr/~pouzet/lucid-synchrone/.

The language was used, from 1996 to 2006 as a laboratory to experiment various extensions of the language Lustre. Several programming constructs (e.g. merge, last, mix of data-flow and control-structures like automata), type-based program analysis (e.g., typing, clock calculus) and compilation methods, originaly introduced in Lucid Synchrone are now integrated in the new SCADE 6 compiler developed at Esterel-Technologies and commercialized since 2008.

Three major release of the language has been done and the current version is V3 (dev. in 2006). The language is still used for teaching and in our research but we do not develop it anymore. Nonetheless, we have integrated several features from Lucid Synchrone in new research prototypes described below.

5.2. ReactiveML

Participants: Mehdi Dogguy, Louis Mandel [contact], Cédric Pasteur.

ReactiveML is a programming language dedicated to the implementation of interactive systems as found in graphical user interfaces, video games or simulation problems. ReactiveML is based on the synchronous reactive model due to Boussinot, embedded in an ML language (Objective Caml).

The Synchronous reactive model provides synchronous parallel composition and dynamic features like the dynamic creation of processes. In ReactiveML, the reactive model is integrated at the language level (not as a library) which leads to a safer and a more natural programming paradigm.

ReactiveML is distributed at URL http://www.lri.fr/~mandel/rml. The compiler is distributed under the terms of the Q Public License and the library is distributed under the terms of the GNU Library General Public License. The development of ReactiveML started at the University Paris 6 (from 2002 to 2006).

The language was mainly used for the simulation of mobile ad hoc networks at the University Paris 6 and for the simulation of sensor networks at France Telecom and Verimag (CNRS, Grenoble).

5.3. Heptagon

Participants: Cédric Pasteur [contact], Brice Gelineau, Léonard Gérard, Adrien Guatto, Cédric Pasteur, Marc Pouzet.

Heptagon is an experimental language for the implementation of embedded real-time reactive systems. It is developed inside the Synchronics large-scale initiative, in collaboration with INRIA Rhones-Alpes. It is essentially a subset of Lucid Synchrone, without type inference, type polymorphism and higher-order. It is thus a Lustre-like language extended with hierarchical automata in a form very close to SCADE 6. The intention for making this new language and compiler is to develop new aggressive optimization techniques for sequential C code and compilation methods for generating parallel code for different platforms. This explains much of the simplifications we have made in order to ease the development of compilation techniques.
Some extensions have already been made, most notably automata. It’s currently used to experiment with linear typing for arrays and also to introduce a concept of asynchronous parallel computations. The compiler developed in our team generates C, java and VHDL code.

Heptagon is jointly developed by Gwenael Delaval and Alain Girault from the INRIA POP ART team (Grenoble).

5.4. Lucy-n

Participants: Louis Mandel [contact], Adrien Guatto, Marc Pouzet.

http://www.lri.fr/~mandel/lucy-n

Lucy-n is a language to program in the n-synchronous model. The language is similar to Lustre with a buffer construct. The Lucy-n compiler ensures that programs can be executed in bounded memory and automatically computes buffer sizes. Hence this language allows to program Kahn networks, the compiler being able to statically compute bounds for all FIFOs in the program.

5.5. ML-Sundials

Participants: Timothy Bourke, Marc Pouzet [contact].

ML-Sundials library provides an Ocaml interface to the Sundials numerical suite 10 (version 2.4.0). This library is used for solving and initial value problem and includes a zero-crossing detection mechanism. Only the CVODE solver with serial nvectors is currently supported. The structure and naming conventions largely follow the original libraries, both for ease of reading the existing documentation and for converting existing source code, but several changes have been made for programming convenience, namely:

- solver sessions are configured through algebraic data types rather than through multiple function calls,
- error conditions are signalled by exceptions rather than return codes (including in user-supplied callback routines),
- closures (partial applications of higher-order functions) are used to share user data between callback routines, and,
- explicit free commands are not necessary nor provided since Ocaml is a garbage-collected language.

The library is in use in a new synchronous hybrid language we are currently developing.

5.6. GCC

Participants: Albert Cohen [contact], Tobias Grosser, Antoniu Pop, Konrad Trifunovic, Feng Li, Riyadh Baghdadi, Cupertino Miranda.

http://gcc.gnu.org

Licence: GPLv3+ and LGPLv3+

The GNU Compiler Collection includes front ends for C, C++, Objective-C, Fortran, Java, Ada, and Go, as well as libraries for these languages (libstdc++, libgcj,...). GCC was originally written as the compiler for the GNU operating system. The GNU system was developed to be 100% free software, free in the sense that it respects the user’s freedom.

PARKAS contributes to the polyhedral compilation framework, also known as Graphite. We also distribute an experimental branch for a stream-programming extension of OpenMP, parallel data-flow programming, and automatic parallelization to a data-flow runtime or architecture. This experiment borrows key design elements to synchronous data-flow languages.

10 https://computation.llnl.gov/casc/sundials/main.html
Tobias Grosser is the maintainer of the Graphite optimization pass of GCC.

5.7. isl

Participants: Sven Verdoolaege [contact], Tobias Grosser, Albert Cohen.

http://freshmeat.net/projects/isl

Licence: LGPLv2.1+

isl is a library for manipulating sets and relations of integer points bounded by linear constraints. Supported operations on sets include intersection, union, set difference, emptiness check, convex hull, (integer) affine hull, integer projection, transitive closure (and over-approximation), computing the lexicographic minimum using parametric integer programming. It also includes an ILP solver based on generalized basis reduction. isl also supports affine transformations for polyhedral compilation.
5. Software


Participants: Bruno Barras [TypiCal team, Saclay], Yves Bertot [Marelle team, Sophia], Frédéric Besson [Lande team, Rennes], Pierre Bouillier, Xavier Clerc [SED team], Pierre Corbineau [University Joseph Fourier, Grenoble], Pierre Courtieu [CNAM], Julien Forest [CNAM], Stéphane Glondu, Benjamin Grégoire [Marelle team, Sophia], Vincent Gross, Hugo Herbelin [correspondant], Stéphane Lescuyer [ProVal team, Saclay], Pierre Letouzey, Assia Mahboubi [TypiCal team, Saclay], Julien Narboux [University of Strasbourg], Jean-Marc Notin [TypiCal team, Saclay], Christine Paulin [Proval team, Saclay], Loïc Pottier [Marelle team, Sophia], Matthias Puech, Yann Régis-Gianas, Vincent Siles, Elie Soubiran, Matthieu Sozeau, Arnaud Spiwack, Pierre-Yves Strub [Formes team, Beijing], Laurent Théry [Marelle team, Sophia], Benjamin Werner [TypiCal team, Saclay].

5.1.1. Version 8.4

Version 8.4 beta was released in December 2011. It introduces a new proof engine designed and implemented by Arnaud Spiwack and a new extensive modular library of arithmetic contributed by Pierre Letouzey. It also includes an extension of the underlying logic with \( \eta \)-conversion by Hugo Herbelin and “commutative-cuts compliant guard condition” by Pierre Bouillier, an extension of the pattern-matching compilation algorithm by Hugo Herbelin, an extension of the procedure of simplification of polynomial expressions by Loïc Pottier, a refinement of the type classes mechanism by Matthieu Sozeau, a new communication model by Vincent Gross for the graphical user interface CoqIDE, that Pierre Letouzey, Pierre Bouillier and Pierre-Marie Pédrot further extended.

Several users gracefully contributed improvements of various features (Tom Prince, Enrico Tassi, Daniel Grayson, Hendrik Tews, ...).

5.1.2. Graphical user interface

Pierre Letouzey has finalized the work initiated by Vincent Gross (former ADT engineer) concerning the CoqIDE user interface: CoqIDE and Coq are now separate unix processes, enhancing the reliability and improving the user experience.

5.1.3. Type inference, tactics, unification and type classes

Matthieu Sozeau corrected important issues with the unification algorithm and enhanced it to support universes. He improved the type-class implementation, adding support for forward-reasoning instances.

To improve the power of induction tactics, Hugo Herbelin added new heuristics for second-order pattern-matching based on ideas from Chung-Kil Hur’s Hseq plugin.

Pierre Letouzey extended the pattern-matching feature of the tactic language.

5.1.4. Internal architecture of the Coq software

Pierre Letouzey also initiated a large reorganization of the internal components of Coq, since these components are currently too much interdependent. This work aims at better isolating components and explicating the interfaces between them. In addition to the initial goal of simplifying the compilation of Coq, having a clearer architecture is also expected to help new contributors when they discover and interact with this large and complex code-base. It also brings new prospects in direct communications between tools developed around Coq. This is a long-term effort that extends beyond the Coq 8.4 release.
Pierre Boutillier worked on the build system generator for Coq users contributions. It now handles correctly developments involving ML files. Files to build developments are also used by CoqIDE to infer the required arguments when it opens a file of a development.

5.1.5. Efficiency

Pierre Letouzey has pursued his effort concerning the improvement of many aspects of the internals of Coq. In particular, with Yann Régis-Gianas, he enabled a faster load of libraries by default, thanks to laziness, and also a better sharing of structures in memory (via better hash-consing), with lower memory footprint and some speedup as visible result. Many bugs have also been addressed.

Starting from September, Xavier Clerc has worked on the codebase in order to profile typical executions. Some hotspots were identified, most notably in comparison functions: some minor modifications led to a gain of a few percents on average. Some tests led to envision the use of a Coq-specialized version of comparison functions, superseding the generic OCaml ones.

5.1.6. General maintenance

Hugo Herbelin, Pierre Letouzey, Pierre Boutillier, Stéphane Glondu and Matthieu Sozeau worked on the general maintenance of the system.

5.1.7. Development Action

A new “Action de Développement Technologique” about Coq has started September 2011. It gathers the \( \pi r^2 \) team, the Marelle team and the CPR team from CNAM, Hugo Herbelin acting as the coordinator. It supports visits and meetings between developers and aims at strengthening the community of Coq users and contributors.

5.1.8. Formalisation in Coq

Stéphane Glondu is working with Mehdi Dogguy on the formalisation in Coq of a type system for a timed asynchronous \( \pi \)-calculus that guarantees confluence.

5.2. Pangolin

Participant: Yann Régis-Gianas.

Yann Régis-Gianas maintained a prototype version of Pangolin. He used it to prove concrete complexity bounds for a set of functional programs using the method described in his FOPARA 2011 paper [16].

5.3. Other software developments

Stéphane Glondu is involved in the maintenance of OCaml-related packages in Debian, which include OCaml itself, Coq, Ssreflect (an extension of Coq developed at INRIA-MSR joint center) and Ocsigen (a web framework developed at PPS). The Ubuntu distribution naturally benefits from this work.

In collaboration with François Pottier (INRIA Gallium), Yann Régis-Gianas maintained Menhir, an LR parser generator for OCaml.
5. Software

5.1. FGb

Participant: J.C. Faugère [contact].

FGb/Gb is a powerful software for computing Gröbner bases; it is written in C/C++ (approximately 250000 lines counting the old Gb software).

5.2. FGb

Participant: Jean-Charles Faugere [correspondant].

FGb is a powerful software for computing Groebner bases. It includes the new generation of algorithms for computing Gröbner bases polynomial systems (mainly the F4, F5 and FGLM algorithms). It is implemented in C/C++ (approximately 250000 lines), standalone servers are available on demand. Since 2006, FGb is dynamically linked with Maple software (version 11 and higher) and is part of the official distribution of this software. See also the web page http://www-salsa.lip6.fr/~jcf/Software/FGb/index.html.

- ACM: I.1.2 Algebraic algorithms
- Programming language: C/C++

5.3. RAGlib

Participant: M. Safey El Din [contact].

RAGLib is a Maple library for computing sampling points in semi-algebraic sets.

5.4. Epsilon

Participant: D. Wang [contact].

Epsilon is a library of functions implemented in Maple and Java for polynomial elimination and decomposition with (geometric) applications.
SECRET Project-Team (section vide)
5. Software

5.1. T-Gems

T-GEMS is a Geometric Kernel for modeling curves and surfaces.

5.2. Ink Simulation in Maya

We have built 3D ink simulation plug-in in Autodesk Maya.
CLASSIC Project-Team (section vide)
GAMMA3 Project-Team (section vide)
4. Software

4.1. PREMIA

Participants: Antonino Zanette, Mathfi Research team, Agnès Sulem [correspondant].

Premia is a software designed for option pricing, hedging and financial model calibration. It is provided with its C/C++ source code and an extensive scientific documentation. https://www-rocq.inria.fr/mathfi/Premia

Thie Premia project keeps track of the most recent advances in the field of computational finance in a well-documented way. It focuses on the implementation of numerical analysis techniques for both probabilistic and deterministic numerical methods. An important feature of the platform Premia is the detailed documentation which provides extended references in option pricing.

Premia is thus a powerful tool to assist Research & Development professional teams in their day-to-day duty. It is also a useful support for academics who wish to perform tests on new algorithms or pricing methods without starting from scratch.

Besides being a single entry point for accessible overviews and basic implementations of various numerical methods, the aim of the Premia project is:

1. to be a powerful testing platform for comparing different numerical methods between each other;
2. to build a link between professional financial teams and academic researchers;
3. to provide a useful teaching support for Master and PhD students in mathematical finance.

• AMS: 91B28;65Cxx;65Fxx;65Lxx;65Pxx
• License: Licence Propriétaire (genuin license for the Consortium Premia)
• Type of human computer interaction: Console, interface in Nsp
• OS/Middleware: Linux, Mac OS X, Windows
• APP: The development of Premia started in 1999 and 13 are released up to now and registered at the APP agency.
• Programming language: C/C++ librairie Gtk
• Documentation: the PNL library is interfaced via doxygen
• Size of the software: 250 Mbyte, 40 Mbyte of C/C++ routines; Number of lines of code: 972000 (for the source part only)
• Publications: [ 1 ] [ 71 ] [ 78 ] [ 86 ] [ 88 ], [ 62 ]

4.1.1. Content of Premia

Premia contains various numerical algorithms (Finite-differences, trees and Monte-Carlo) for pricing vanilla and exotic options on equities, interest rate, credit and energy derivatives.

1. Equity derivatives:

The following models are considered:

Black-Scholes model (up to dimension 10), stochastic volatility models (Hull-White, Heston, Fouque-Papanicolaou-Sircar), models with jumps (Merton, Kou, Tempered stable processes, Variance gamma, Normal inverse Gaussian), Bates model.
For high dimensional American options, Premia provides the most recent Monte-Carlo algorithms: Longstaff-Schwartz, Barraquand-Martineau, Tsitsklis-Van Roy, Broadie-Glassermann, quantization methods and Malliavin calculus based methods.

Dynamic Hedging for Black-Scholes and jump models is available.
Calibration algorithms for some models with jumps, local volatility and stochastic volatility are implemented.

2. **Interest rate derivatives**
The following models are considered:
Premia provides a calibration toolbox for Libor Market model using a database of swaptions and caps implied volatilities.

3. **Credit derivatives: CDS, CDO**
Reduced form models and copula models are considered.
Premia provides a toolbox for pricing CDOs using the most recent algorithms (Hull-White, Laurent-Gregory, El Karoui-Jiao, Yang-Zhang, Schönbucher)

4. **Hybrid products**
PDE solver for pricing derivatives on hybrid products like options on inflation and interest or change rates is implemented.

5. **Energy derivatives: swing options**
Mean reverting and jump models are considered.
Premia provides a toolbox for pricing swing options using finite differences, Monte-Carlo Malliavin-based approach and quantization algorithms.

### 4.1.2. Development of the PNL Library
Here are the major contribution of J. Lelong:

1. Development of the PNL.
   - A PnlArray object has been added to create arrays of PnlObjects.
   - The implementation of the PnlList type has been changed to improve linear iteration on a list.
   - Design of a new unit test framework and backport of all the previous tests.
   - Integration of the Runge Kutta Fehler 45 method for solving n dimensional ODEs.
   - New organisation of the manual.
   - Random number generators: new Sobol generators (32 and 64 bits), dynamic Mersenne Twister updated to version 0.6.1. All random number generators are now thread-safe.
   - Update of internal Lapack (and the corresponding shipped version of Blas) to version 3.2.1
   - Update of F2C.
   - LU factorization for tridiagonal matrices.
   - Cholesky block factorization for positive semi-definite matrices.

### 4.1.3. Premia design
Anton Kolotaev (ADT engineer), supervised by J. Lelong, has realized the Bindings for Premia with Python and F#; a library allowing the development of bindings to other languages; a Web interface Web for Premia.
Moreover he has improved the documentation facility system of Premia adn has updated the Excel interface Excel for the new versions MS Excel and MS Windows.

Tasks achieved by J. Lelong:

1. New design of the enumeration type to allow the number of parameters of Premia objects to depend on the selected value within the enumeration. This change had a strong impact on the VAR system and many core functions had to be rewritten. This modification was definitely essential to improve the Nsp interface, which was broken for long as far as credit derivatives are concerned and is now working properly again.

2. Creation of the first Premia bundle for Mac OS X and automation of the building process.

3. Generic “Get”, “FGet” and “Print” functions had been introduced to simplify object creation but there were still many exceptions which were not using these generic functions. All these exceptions have been handled and now rely on the generic functions.

4. Improvement of the scripts to build the free version of Premia.

5. Integration of Cosine methods implemented by Bowen Zhang.

Tasks achieved by C. Labart:

- Improvement of the credit part of Premia: correction of memory leaks, modification of old codes to use the new copula structure. This enables to remove a large part of the code which has become hard to maintain.
- Complete rewriting of the BSDE algorithm for pricing basket options in high dimension to use a more efficient approximation technique. This has been possible thanks to the polynomial approximation tool provided by the PNL.

**4.1.4. Algorithms implemented in Premia in 2011**

Premia 13 was delivered to the consortium members in March 2011. It contains the following new algorithms:

**4.1.4.1. Interest rate derivatives**

- Pricing and hedging callable Libor exotics in forward Libor models. V. Piterbarg, *The Journal of Computational Finance* Volume 8 Number 2, Winter 2004/05

**4.1.4.2. Credit risk derivatives**


**4.1.4.3. Electricity derivatives**

- Variance optimal hedging for processes with independent increments and applications. Applications to electricity market. F.Russo, S. Goutte and N. Oudjane. *Preprint 2010*
4.1.4.4. Equity derivatives

- Pricing options under stochastic volatility: a power series approach. F.Antonelli and S.Scarlatti *Finance & Stochastics*, Volume XIII (2009), issue 1
- Monte Carlo for pricing Asian options in jump models. E. Dia and D. Lamberton, *Preprint*
- Doubly Reflected BSDEs with Call Protection and their Approximation. J.F. Chassagneux S.Crepey. *Preprint* 2010

4.1.5. New algorithms for the release 14 of Premia to be delivered in March 2012 to the Consortium:

4.1.5.1. Interest rate derivatives


4.1.5.2. Credit risk derivatives
• Calibration in a local and stochastic intensity model. A. Alfonsi, C. Labart and J. Lelong, *Preprint*.

4.1.5.3. Energy and commodities

• Markov Models for Commodity Futures: Theory and Practice. L. Andersen. *Preprint*

4.1.5.4. Equity derivatives

• S. Ould-Aly Revised Bergomi model. *Preprint*
• Volatility derivatives in market models with jumps. H. Lo A. Mijatovic, *Preprint*.
• High-order discretization for stochastic correlation models. A. Alfonsi and A. Ahdida *Preprint*
• Fast and Accurate Long Stepping Simulation of the Heston Stochastic Volatility Model J.H. Chan, M.S. Joshi, *Preprint*
• Ninomiya-Victoir scheme for variance swaps model.
• Multilevel adaptive Monte Carlo. A. Kebaier, K. Hajji
• Wiener-Hopf techniques for Path-dependent Options in Bates and Heston model. Kudryavtsev O.
• Continuously monitored barrier options under Markov processes. A. Mijatovic, M. Pistorius, preprint.
• Exotic derivatives in a dense class of stochastic volatility models with jumps. A. Mijatovic and M. Pistorius, *Preprint*.
• Pricing Discretely Monitored Asian Options by Maturity Randomization. G. Fusai, D. Marazzina and M. Marena. *Preprint*
• B. Lapeyre and A. Abbas-Turki American Options Based on Malliavin Calculus and nonparametric Variance Reduction Methods on Malliavin Calculus. *Preprint*.
• Backward Stochastic Differential Equations (BSDE).
• American options in high dimension solving BSDE with penalization.
MICMAC Project-Team (section vide)
5. Software

5.1. Introduction

We are led to develop two types of software. The first category is prototype softwares: various softwares are developed in the framework of specific research contracts (and sometimes sold to the contractor) or during PhD theses. They may be also contributions to already existing softwares developed by other institutions such as CEA, ONERA or EDF. The second category is advanced software which are intended to be developed, enriched and maintained over longer periods. Such softwares are devoted to help us for our research and/or promote our research. We have chosen to present here only our advanced softwares.

5.2. MELINA

This software has been developed under the leadership of D. Martin for several years in order to offer to the researchers a very efficient tool (in Fortran 77 and object oriented) for easily implementing finite element based original numerical methods for solving partial differential equations. It has specific and original potential in the domain of time harmonic wave problems (integral representations, spectral DtN conditions,...). Nowadays, it is fully functional in various application areas (acoustics and aeroacoustics, elastodynamics, electromagnetism, water waves). It is an open source software with on line documentation available at http://homepage.mac.com/danielmartin/melina/

The software is regularly used in about 10 research laboratories (in France and abroad) and number of research papers have published results obtained with MELINA (see the Web site). Moreover, every 2 years, a meeting is organized which combines a workshop which teaches new users with presentations by existing users.

During the last four years, apart from various local improvements of the code, new functionalities have been developed:

- Higher order finite elements (up to 10th order),
- Higher order quadrature formulae,
- DtN boundary conditions in 3D.

A new C++ version of the software is under development.

5.3. MONTJOIE

Montjoie is a software for the efficient and accurate wave propagation numerical modeling in both time dependent or time harmonic regimes in various domains of application: acoustics, aeroacoustics, elastodynamics and electromagnetism. It is based essentially on the use of hexahedral-dominant (including a small part of tetrahedra, pyramids and prisms) conforming meshes and continuous or discontinuous Galerkin approximations. The use of tensor product basis functions coupled to appropriate numerical quadrature techniques leads to important gains in both computing time and memory storage. Various techniques for treating unbounded domains have been incorporated: DtN maps, local absorbing conditions, integral representations and PMLs.

We have written an interface for the use of other libraries: SELDON, a C++ linear algebra library (interfaced with BLAS and LAPACK) used for iterative linear solvers, MUMPS, PASTIX and UMFPACK for sparse direct solvers, ARPACK for eigenvalue computation, METIS and SCOTCH for mesh decomposition. Except for trivial geometries, the mesh generation is not part of the code. It can be done with Modulef, Gmsh, Ghs3D or Cubit.
This code has been developed by Marc Duruflé during his PhD thesis (in 2006). Some other contributors have brought more specific enrichments to the code. The online documentation is available at:

http://montjoie.gforge.inria.fr/.

The main contributions of 2010 have been the following:

- parallelization of the interface with ARPACK,
- unitary tests for SELDON and Montjoie aiming at stabilizing the code,
- implementation of $H(div)$ finite elements in 2D and 3D in collaboration with Morgane Bergot,
- optimization of the numerical simulation of the piano in collaboration with Juliette Chabassier (multithreading for the string nonlinear equations, improvement of the parallelization).

5.4. XLIFE++

During 2011, we performed a deep analysis of the two finite elements software developed by the lab (Melina and Montjoie) in order to propose a new software in C++ with extended capabilities and more integrated tools. The results of this analysis lead us to keep the general philosophy of Melina software (unified variational approach) but with major evolutions: integrated meshing tools, new variational description allowing FEM, BEM and DG formulation in an unified framework, global and local finite element computations, new approach to take into account essential boundary conditions and high performance computing skills (multithread and GPU computation). This new development will partly be supported by the Simposium european project dedicated to Non Destructive Testing tools (leader CEA/LIST, from september 2011 to august 2014) which requires some numerical simulation tools such as finite element library. It also a collaborative project with IRMAR (Rennes University). This new library, named xlife++ for eXtended LIbrary of Finite Element in C++, is an open source library (LGPL license) and its repository is the INRIA Gforge.
5. Software

5.1. inlinedocs

Participant: Toby Hocking [correspondant].

Generates Rd files from R source code with comments, providing for quick, sustainable package development. The syntax keeps code and documentation close together, and is inspired by the Don’t Repeat Yourself principle. See also the web page http://inlinedocs.r-forge.r-project.org/.

- Version: 1.8
- Contact: toby.hocking@inria.fr

5.2. directlabels

Participant: Toby Hocking [correspondant].

The directlabels package provides an extensible framework for automatically placing direct labels onto multicolor lattice or ggplot2 plots. It includes heuristics for examining "lattice" and "ggplot" objects and inferring an appropriate Positioning Method for placing the labels. Furthermore, the design of directlabels makes it simple to create Positioning Methods for specific plots or libraries of portable Positioning Methods that can be re-used. See also the web page http://directlabels.r-forge.r-project.org/.

- Version: 2.2
- Contact: toby.hocking@inria.fr

5.3. clusterpath

Participant: Toby Hocking [correspondant].

The clusterpath package provides an R/C++ implementation of the algorithms described in [16]. See also the web page http://clusterpath.r-forge.r-project.org/.

- Version: 1.0
- Contact: toby.hocking@inria.fr

5.4. UGM

Participant: Mark Schmidt [correspondant].

UGM is a set of Matlab functions implementing various tasks in probabilistic undirected graphical models of discrete data with pairwise (and unary) potentials. Specifically, it implements a variety of methods for the following four tasks:

- Decoding: Computing the most likely configuration.
- Inference: Computing the partition function and marginal probabilities.
- Sampling: Generating samples from the distribution.
- Parameter Estimation: Given data, computing maximum likelihood (or MAP) estimates of the parameters.
The first three tasks are implemented for arbitrary discrete undirected graphical models with pairwise potentials. The last task focuses on Markov random fields and conditional random fields with log-linear potentials. The code is written entirely in Matlab, although more efficient mex versions of some parts of the code are also available.

See also the web page http://www.di.ens.fr/~mschmidt/Software/UGM.html.

- Version: 2011
- Contact: mark.schmidt@inria.fr

5.5. alphaBeta

Participant: Mark Schmidt [correspondant].

The code contains implementations of several available methods for the problem of computing an approximate minimizer of the sum of a set of unary and pairwise real-valued functions over discrete variables. This equivalent to the problem of MAP estimation, also known as decoding, in a pairwise undirected graphical model. The code focuses on scenarios where the pairwise energies encourage neighboring variables to take the same state. The particular methods contained in the package are iterated conditional mode, alpha-beta swaps, alpha-expansions, and alpha-expansion beta-shrink moves.

See also the web page http://www.di.ens.fr/~mschmidt/Software/alphaBeta.html.

- Version: 1
- Contact: mark.schmidt@inria.fr

5.6. Matlab Software from “Graphical Model Structure Learning with L1-Regularization”

Participant: Mark Schmidt [correspondant].

This package contains the code used to produce the results in Mark Schmidt’s thesis: Roughly, there are five components corresponding to five of the thesis chapters:

- Chapter 2: L-BFGS methods for optimizing differentiable functions plus an L1-regularization term.
- Chapter 3: L-BFGS methods for optimizing differentiable functions with simple constraints or regularizers.
- Chapter 4: An L1-regularization method for learning dependency networks, and methods for structure learning in directed acyclic graphical models.
- Chapter 5: L1-regularization and group L1-regularization for learning undirected graphical models, using either the L2, Linf, or nuclear norm of the groups.
- Chapter 6: Overlapping group L1-regularization for learning hierarchical log-linear models, and an active set method for searching through the space of higher-order groups.

See also the web page http://www.di.ens.fr/~mschmidt/Software/thesis.html.

- Version: 1
- Contact: mark.schmidt@inria.fr

5.7. Hybrid deterministic-stochastic methods for data fitting

Participant: Mark Schmidt [correspondant].
Many structured data-fitting applications require the solution of an optimization problem involving a sum over a potentially large number of measurements. Incremental gradient algorithms (both deterministic and randomized) offer inexpensive iterations by sampling only subsets of the terms in the sum. These methods can make great progress initially, but often slow as they approach a solution. In contrast, full gradient methods achieve steady convergence at the expense of evaluating the full objective and gradient on each iteration. We explore hybrid methods that exhibit the benefits of both approaches. Rate of convergence analysis and numerical experiments illustrate the potential for the approach.


- Version: 1
- Contact: mark.schmidt@inria.fr
- Participants outside of Sierra: Michael Friedlander (Scientific Computing Laboratory, Department of Computer Science, University of British Columbia)

### 5.8. Multi-task regression using minimal penalties

**Participant:** Matthieu Solnon [correspondant].

This toolbox implements statistical algorithms designed to perform multi-task kernel ridge regressions, as described in [33].

See also the web page http://www.di.ens.fr/~solnon/articles/multi-task_regression/multitask_minpen_en.html.

- Version: 1
- Contact: matthieu.solnon@ens.fr
4. Software

4.1. Software

4.1.1. Continuation of M3N

A large part of the software currently in use in the project-team was initiated and developed within former projects (Menusin, M3N).

4.1.2. CellSys

Participants: Dirk Drasdo [correspondent], Stefan Höhme [Research Associate, University of Leipzig], Adrian Friebel [PhD student, University of Leipzig], Tim Johann [Software Engineer, University of Leipzig], Nick Jagiella [PhD student].

Computer simulation software for individual cell (agent)-based models of tumour and tissue growth solved either by systems of coupled equations of motion for each individual cell or by Kinetic Monte Carlo methods [77].
5. Software

5.1. Urban air quality analysis

Participants: Anne Tilloy, Vivien Mallet.

“Urban Air Quality Analysis” carries out data assimilation at urban scale. It merges the outputs of a numerical model (maps of pollutant concentrations) with observations from an air quality monitoring network, in order to produce the so-called analyses, that is, corrected concentration maps. The data assimilation computes the Best Linear Unbiased Estimator (BLUE), with a call to the data assimilation library Verdandi. The error covariance matrices are parameterized for both model simulations and observations. For the model state error covariances, the parameterization primarily relies on the road network. The software handles ADMS output files, for a posteriori analyses or in an operational context.

5.2. Polyphemus

Participants: Vivien Mallet, Pierre Tran, Damien Garaud, Anne Tilloy.

Polyphemus (see the web site http://cerea.enpc.fr/polyphemus/) is a modeling system for air quality. As such, it is designed to yield up-to-date simulations in a reliable framework: data assimilation, ensemble forecast and daily forecasts. Its completeness makes it suitable for use in many applications: photochemistry, aerosols, radionuclides, etc. It is able to handle simulations from local to continental scales, with several physical models. It is divided into three main parts:

- libraries that gather data processing tools (SeldonData), physical parameterizations (AtmoData) and postprocessing abilities (AtmoPy);
- programs for physical preprocessing and chemistry-transport models (Polair3D, Castor, two Gaussian models, a Lagrangian model);
- drivers on top of the models in order to implement advanced simulation methods such as data assimilation algorithms.

Figure 1 depicts a typical result produced by Polyphemus. Clime is involved in the overall design of the system and in the development of advanced methods in model coupling, data assimilation and ensemble forecast (through drivers and post-processing).

In 2011, Polyphemus was extended for a better integration with the data assimilation library Verdandi. A first (unstable) version of Polyphemus with a complete overhaul of the input/output operations and of the configuration files was provided to the developers. The derivative of Polyphemus that is used at IRSN was used for the first time in a crisis context in order to simulate the transport of radionuclides during the Fukushima nuclear disaster.

5.3. Data assimilation library: Verdandi

Participants: Kévin Charpentier, Marc Fragu [MACS], Vivien Mallet, Dominique Chapelle [MACS], Philippe Moireau [MACS], Sergiy Zhuk, Anne Tilloy.

The leading idea is to develop a data assimilation library intended to be generic, at least for high-dimensional systems. Data assimilation methods, developed and used by several teams at INRIA, are generic enough to be coded independently of the system to which they are applied. Therefore these methods can be put together in a library aiming at:

- making easier the application of methods to a great number of problems,
- making the developments perennial and sharing them,
- improving the broadcast of data assimilation works.
An object-oriented language (C++) has been chosen for the core of the library. A high-level interface to Python is automatically built. The design raised many questions, related to high dimensional scientific computing, the limits of the object contents and their interfaces. The chosen object-oriented design is mainly based on three class hierarchies: the methods, the observation managers and the models. Several base facilities have also been included, for message exchanges between the objects, output saves, logging capabilities, computing with sparse matrices.

In 2011, versions 0.9, 1.0 and 1.1 of Verdandi were released. These versions are advanced enough to be used by the data assimilation community. Compared to previous versions, the additions are: 4D-Var, ensemble Kalman filter, redesigned perturbation managers, sequential aggregation, improvements in the documentation and an improved support of Windows.
3. Software

3.1. M1cg1
   - Participant: J. Ch. Gilbert.
   - Version: 1.2.
   - Programming language: Fortran 77.
   - Solves a convex quadratic optimization problem and builds a preconditioning matrix, 1 download in 2011.
   - See also the web page http://www-roc.inria.fr/~gilbert/modulopt/optimization-routines/m1cg1/m1cg1.html.

3.2. M1qn3
   - Participants: J. Ch. Gilbert, Cl. Lemaréchal.
   - Version: 3.3.
   - Programming language: Fortran 77.
   - Solves a very large scale differentiable optimization problem, 45 downloads in 2011.
   - See also the web page http://www-roc.inria.fr/~gilbert/modulopt/optimization-routines/m1qn3/m1qn3.html.

3.3. PHlab
   - Participant: J. Ch. Gilbert.
   - Version: 0.1.
   - Programming language: Matlab.
   - Solves a stochastic linear optimization problem defined on a scenario tree by the Progressive Hedging algorithm [12].

3.4. Sklml
   Participants: Quentin Carbonneaux, François Clément, Pierre Weis.
   Easy coarse grain parallelization.
   See also the web page http://sklml.inria.fr/.
   - Version: 1.0+pl1
   - Programming language: OCaml

3.5. SQPlab
   - Participant: J. Ch. Gilbert.
   - Version: 0.4.5.
   - Programming language: Matlab.
   - See also the web page http://www-roc.inria.fr/~gilbert/modulopt/optimization-routines/sqplab/sqplab.html.

3.6. LifeV
   Participant: Michel Kern.
Finite element library with emphasis on life and environmental sciences. LifeV is the joint collaboration between École Polytechnique Fédérale de Lausanne (Switzerland), Politecnico di Milano (Italy), Inria (France) and Emory University (U.S.A.).

- Version 2.0
- Programming language: C++

### 3.7. SOPRANO_scenarios

- Participant: A. Chiche, J. Ch. Gilbert, M. Porcheron
- Version: 0.1.
- Programming language: C++.
- Solves the medium-term electricity planning problem defined on a scenario tree by the Progressive Hedging algorithm.
5. Software

5.1. FELISCE

**Participants:** Dominique Chapelle, Jérémie Foulon [correspondant], Philippe Moireau, Marina Vidrascu.

FELISCE – standing for “Finite Elements for LIfe SCiences and Engineering” – is a new finite element code which the MACS and REO teams have decided to jointly develop in order to build up on their respective experiences concerning finite element simulations. One specific objective of this code is to provide in a unified software environment all the state-of-the-art tools needed to perform simulations of the complex cardiovascular models considered in the two teams – namely involving fluid and solid mechanics, electrophysiology, and the various associated coupling phenomena. FELISCE is written in C++, and may be later released as an opensource library. [https://gforge.inria.fr/projects/felisce/](https://gforge.inria.fr/projects/felisce/)

5.2. HeartLab

**Participants:** Matthieu Caruel, Radomir Chabiniok, Dominique Chapelle, Alexandre Imperiale, Philippe Moireau [correspondant].

The heartLab software is a library written in Matlab and C (mex functions) designed to perform both simulation and estimation (based on various types of measurements, e.g. images) of the heart mechanical behavior. Started in 2006, it is already quite large (about 60,000 lines), and is used within the CardioSense3D community.

The code relies on OpenFEM for the finite element computations, and the implementation was performed with a particular concern for modularity, since modeling and estimation use the same finite element operators. This modularity also allows to couple the code with other FEM solvers, such as LifeV and Mistral developed in the Reo team-project. In particular, we are now able to include perfusion and electrical coupling with LifeV using PVM, and fluid-structure interaction using Mistral.

We also included geometric data and tools in the code to define heart anatomical models compatible with the simulation requirements in terms of mesh quality, fiber direction data defined within each element, and referencing necessary for handling boundary conditions and estimation, in particular. These geometries are analytical or come from computerized tomography (CT) or magnetic resonance (MR) image data of humans or animals.

We recently incorporated numerous non-linear data assimilation observation operators based on medical imaging post-processing to be able to now perform estimation with a large variety of medical imaging modalities.

The Library is now 64 bits compatible with the help of the Cesare Corrado from Reo.

5.3. MITCNL

**Participants:** Dominique Chapelle, Marina Vidrascu [correspondant].

The package MITCNL is a set of subroutines that implements the triangular MITC3, MITC6 and quadrilateral MITC4 and MITC9 shell elements for large displacements [14]. We use it as a basis for new developments of shell elements, in particular within Modulef. It can be easily interfaced with most finite element codes as well. We also license this package to some of our partners for use with their own codes.

5.4. MODULEF

**Participant:** Marina Vidrascu [correspondant].
Most of the software developed in our team is integrated in the Modulef library. Modulef is designed to provide building blocks for effective and reliable software development in finite element analysis. Well-adapted rigorous data structures and ease of integration (for new methods or algorithms) are some of its key advantages. Until 1998, Modulef was distributed by the Simulog company within a club structure (for a membership fee). In order to encourage its dissemination, its status was then changed to make it freely available. It can be downloaded at no charge from the INRIA-Rocquencourt web site (http://www-rocq.inria.fr/modulef/).

5.5. OpenFEM: a Finite Element Toolbox for Matlab and Scilab

Participants: Dominique Chapelle, Philippe Moireau [correspondant].

OpenFEM (http://www.openfem.net) is an opensource finite element toolbox for linear and nonlinear structural mechanics within the Matlab and Scilab matrix computing environments. This software is developed in a collaboration between Macs and the SDTools company. Performing finite element analyses within a matrix computing environment is of considerable interest, in particular as regards the ease of new developments, integration of external software, portability, post-processing, etc.

This Library is the core of the finite element computations of HeartLab where a specific version have been developed with the help of Cesare Corrado from Reo.

5.6. SHELDON

Participants: Dominique Chapelle, Marina Vidrascu [correspondant].

SHELDON (SHELls and structural Dynamics with DOmain decomposition in Nonlinear analysis) is a finite element library based on the Modulef package which contains shell elements, nonlinear procedures and PVM subroutines used in domain decomposition or coupling methods.

5.7. Verdandi

Participants: Dominique Chapelle, Marc Fragu [correspondant], Vivien Mallet, Philippe Moireau.

Verdandi is an opensource (LGPL) software library aiming at providing assimilation data methods and related tools. Mainly targeted at large systems arising from the discretization of PDEs, it is intentionally devised as generic, which allows for applications in a wide range of problems (biology and medicine, environment, image processing...). See also the web page http://verdandi.gforge.inria.fr/, with a complete documentation in English. The first stable version (1.0) was released in June and contains most of the major data assimilation algorithms of both variational and sequential types. Moreover, some specific developments are performed with particular regard to cardiac modeling applications, as Verdandi is partly funded by – and distributed within – the euHeart project.

- ACM: Mathematical software
- AMS: System theory; control
- Software benefit: Verdandi est la seule bibliothèque d’assimilation de données générique.
- License: LGPL (2.1 or any later version)
- Type of human computer interaction: Ligne de commande et fichiers de configuration
- OS/Middleware: Linux, MacOS ou Windows
- Required library or software: Seldon (LGPL, http://seldon.sourceforge.net/)
- Documentation: Chaque fonction est documentée, grâce à Doxygen. Il y a aussi un guide d’utilisation (en cours de rédaction actuellement). Toute la documentation est en anglais.

1 http://www.sdtools.com
4. Software

4.1. Virtual Retina: A Large-Scale Simulator of Biological Retina

Participants: Bruno Cessac, Hassan Nasser, Pierre Kornprobst [correspondent], Adrien Wohrer [Group for Neural Theory - ENS].

Virtual Retina is a simulation software developed by Adrien Wohrer during his PhD [74], [73] that allows large-scale simulations of biologically-plausible retinas.

Virtual Retina has a variety of biological features implemented such as (i) spatio-temporal linear filter implementing the basic center/surround organization of retinal filtering, (ii) non-linear contrast gain control mechanism providing instantaneous adaptation to the local level of contrast; (iii) spike generation by one or several layers of ganglion cells paving the visual field.

Virtual Retina is under INRIA CeCill C open-source licence, so that one can download it, install it and run it on your own sequences. Virtual Retina also offers a web service (v 2.1), so that you may test directly the main software on user's own data, without any installation. This webservice was developed in collaboration with Nicolas Debeissat (engineer, 2002).

Virtual Retina continues its evolution thanks to the work done in our team by Bruno Cessac and Hassan Nasser who are interested in the analysis of the collective behavior of ganglion cells responses (see Section 5.1.1).

To take this collective behavior into account, Virtual Retina needs to be extended since in its current version, ganglion cells are independent. Other evolutions of Virtual Retina are also investigated by external partners (see, e.g., [68].

- IDDN number: IDDN.FR.001.210034.000.S.P.2007.000.31235
- Version: v 2.2.2 (September 2011)
- Link: http://www-sop.inria.fr/neuromathcomp/public/software/virtualretina

4.2. ABFilter: A Simulator Of V1 Simple and Complex Cells

Participants: Pierre Kornprobst [correspondant], Maria-Jose Escobar [Electronics Engineering Department of the Universidad Técnica Federico Santa María, Valparaíso, Chile.], Adrien Wohrer [Group for Neural Theory - ENS].

ABFilter is a C++-library that allows the implementation of spatiotemporal filtering in video sequences using filters proposed by Adelson and Bergen in [62].

Simple cells in V1 are characterized by linear receptive fields where the neuron response is a weighted linear combination of the input stimulus inside its receptive field. By combining two simple cells in a linear manner it is possible to get direction-selective cells. The direction-selectivity refers to the property of a neuron to respond to the direction of the stimulus motion. The way to model this selectivity is by obtaining receptive fields oriented in space and time. Some characteristics of V1 complex cells can be explained using a nonlinear combination of V1 simple cells as it has been proposed by, e.g., Adelson and Bergen in [62]. Implementing these cells properly is a difficult problem and this library offers the possibility to easily implement a V1 layer which can serve as an input to subsequent cortical areas such as MT (see, e.g., the architecture developed in [65]).

The ABFilter library is under a CeCill-C open-source license.

- IDDN.FR.001.280017.000.S.P.2011.000.31235
- Version: v 1.0 (May 2011)
- Download: http://www-sop.inria.fr/neuromathcomp/public/software/abfilter-1.0.tar.gz
4.3. MotionLib: A Neural-Fields Model for Motion Estimation

Participants: Pierre Kornprobst [correspondent], Emilien Tlapale.

MotionLib implements the neural field model of motion estimation described in [52], using the Python programming language.

Motion integration is the core of the model: It implements a two-layer model with feedbacks that selects and diffuses motion signals. The main structure has been implemented here, allowing contributors to enrich this model easily. Several tools are also provided to visualize and analyze the distributed velocity fields obtained by this approach.

- License IDDN.FR.001.210029.000.S.P.2011.000.31235
- Version: v 1.0 (October 2011)
- Download: http://www-sop.inria.fr/neuromathcomp/public/software/motion.zip

4.4. Event neural assembly Simulation

Participants: Frederic Alexandre [INRIA Cortex Nancy], Bruno Cessac [correspondent], Rodrigo Cofre Torres, Jeremy Fix [INRIA Cortex Nancy], Olivier Rochel [INRIA Cortex Nancy], Sélim Kraria, Olivier Marre, Hassan Nasser, Horacio Rostro-Gonzalez, Vivien Robinet, Thierry Viéville [INRIA Cortex Nancy], Juan-Carlos Vasquez.

Enas is a library providing numerical tools for the simulation of neural networks and the analysis of spike trains either coming from neural simulators or from biological experiments.

It is designed mainly as

- An existing simulator plug-in (e.g. MVASpike or other simulators via the NeuralEnsemble meta-simulation platform),
- Additional modules for computations with neural unit assembly on standard platforms (e.g. Python, Matlab or the Scilab platform).
- Original modules for the analysis of spike train statistics intended to be used by the neuroscientists community.

Achievements include:

- Spike trains statistical analysis via Gibbs distributions. They are based on the estimation of a parametric Gibbs potential optimaly characterizing the statistics of empirical spike trains (by minimisation of the Kullback-Leibler divergence between the empirical measure and the Gibbs measure). From this, classical statistical indicators such as firing rate, correlations, higher order moments and statistical entropy are obtained. Also, the form of the Gibbs potential provides essential informations on the underlying neural network and its structure. This method does not only allows us to estimate the spikes statistics but also to compare different models, thus answering such questions about the neural code as: are correlations (or time synchrony or a given set of spike patterns, ... ) significant with respect to rate coding?
- Spiking network programing for exact event’s sequence restitution;
- Discrete neural field parameters algorithmic adjustments and time-constrained event-based network simulation reconciling clock and event based simulation methods.

Compared to existing libraries Enas offers new computational methods taking into account time constraints in neural networks (such as memory effects), based on theoretical methods rooted in statistical physics and applied mathematics. The algorithms used are based on linear programming, nonlinear parameter estimations, statistical methods. The C/C++ code has been organized as “bean java” to ease its use by programmers non specialized in advanced object programming. As a consequence the code is distributed in the form of an include source for the lightest and the most universal integration into users codes. The standard algorithms are based on the best free libraries in the domain such as gsl http://www.gnu.org/software/gsl.
Event neural assembly simulation is developed in gForge. It is under CeCILL C licence

\textit{APP logiciel Enas: IDDN.FR.OO1.360008.000.S.P2009.000.10600.}

Its development as a friendly software designed for the neuroscience community is our next purpose (ADT proposal).

Website: \url{http://enas.gforge.inria.fr/}
5. Software

5.1. LiFE-V library
Participants: Miguel Ángel Fernández Varela [correspondant], Jean-Frédéric Gerbeau.

LiFE-V\(^2\) is a finite element library providing implementations of state-of-the-art mathematical and numerical methods. It serves both as a research and production library. LiFE-V is the joint collaboration between three institutions: Ecole Polytechnique Fédérale de Lausanne (CMCS) in Switzerland, Politecnico di Milano (MOX) in Italy and INRIA (REO) in France. It is a free software under LGPL license.

5.2. Mistral library
Participants: Cristóbal Bertoglio Beltran, Jean-Frédéric Gerbeau [correspondant], Vincent Martin, Joaquín-Alejandro Mura Mardones.

Mistral is a finite element library which implements in particular fluid-structure interaction algorithms (ALE and Fictitious domain formulations), fluid surface flow (ALE) and incompressible magnetohydrodynamics equations. Mistral results from a collaboration between INRIA and ENPC (CERMICS).

5.3. FELiScE
Participants: Grégory Arbia, Cesare Corrado, Miguel Ángel Fernández Varela, Justine Fouchet-Incaux, David Froger, Jean-Frédéric Gerbeau [correspondant], Damiano Lombardi, Elisa Schenone, Saverio Smal-done.

FELiScE – standing for “Finite Elements for Life Sciences and Engineering” – is a new finite element code which the MACS and REO project-teams have decided to jointly develop in order to build up on their respective experiences concerning finite element simulations. One specific objective of this code is to provide in a unified software environment all the state-of-the-art tools needed to perform simulations of the complex cardiovascular models considered in the two teams – namely involving fluid and solid mechanics, electrophysiology, and the various associated coupling phenomena. FELISCE is written in C++, and may be later released as an opensource library. https://gforge.inria.fr/projects/felisce/

\(^2\) http://www.lifev.org/
4. Software

4.1. The Matlab System Identification ToolBox (SITB)

**Participant:** Qinghua Zhang.

*This development is made in collaboration with Lennart Ljung (Linköping University, Sweden), Anatoli Juditsky (Joseph Fourier University, France) and Peter Lindskog (NIRA Dynamics, Sweden).*

The System Identification ToolBox (SITB) is one of the main Matlab toolboxes commercialized by The Mathworks. INRIA participates in the development of its extension to the identification of nonlinear systems which is released since 2007. It includes algorithms for both black box and grey box identification of nonlinear dynamic systems. INRIA is mainly responsible for the development of black box identification, with nonlinear autoregressive (NLARX) models and block-oriented (Hammerstein-Wiener) models.

4.2. Inverse Scattering for Transmission Lines (ISTL)

**Participants:** Michel Sorine, Qinghua Zhang.

ISTL is a software for numerical computation of the inverse scattering transform for electrical transmission lines. In addition to the inverse scattering transform, it includes a numerical simulator generating the reflection coefficients of user-specified transmission lines. With the aid of a graphical interface, the user can interactively define the distributed characteristics of a transmission line. This software is mainly for the purpose of demonstrating a numerical solution to the inverse problem of non uniform transmission lines. Its current version is limited to the case of lossless transmission lines. It is registered at Agence pour la Protection des Programmes (APP) under the number IDDN.FR.001.120003.000.S.P.2010.000.30705.

4.3. CGAO: Contrôle Glycémique Assisté par Ordinateur

**Participants:** Alexandre Guerrini, Michel Sorine.

This development is made in collaboration with Pierre Kalfon (Chartres Hospital) and Gaëtan Roudillon (LK2).

This software developed with LK2 and Hospital Louis Pasteur (Chartres) provides efficient monitoring and control tools that will help physicians and nursing staff to avoid hyperglycaemia and hypoglycaemia episodes in Intensive Care Units. It is used in a large clinical study, CGAO-REA. Commercialization will be done by LK2.

The software is designed to assist physicians to deal with a variant of the classical Stability/Precision dilemma of control theory met during blood-glucose control. It has been tested in the ICU of Chartres and, since November 2009, it is used in a large scale study launched by the SFAR (French Society of Anesthesia and Intensive Care) involving 62 ICUs and including 6422 patients.

More than 3500 patients have been included in CGAO-REA.

4.4. LARY_CR: Software package for the Analysis of Cardio Vascular and Respiratory Rhythms

**Participants:** Claire Médigue, Serge Steer.
LARY_CR is a software package dedicated to the study of cardiovascular and respiratory rhythms [77]. It presents signal processing methods, from events detection on raw signals to the variability analysis of the resulting time series. The events detection concerns the heart beat recognition on the electrocardiogram, defining the RR time series, the maxima and minima on the arterial blood pressure defining the systolic and diastolic time series. These detections are followed by the resampling of the time series then their analyse. This analyse uses temporal and time frequency methods: Fourier Transform, spectral gain between the cardiac and blood pressure series, Smooth Pseudo Wigner_Ville Distribution, Complex DeModulation, temporal method of the cardiovascular Sequences. The objective of this software is to provide some tools for studying the autonomic nervous system, acting in particular in the baroreflex loop; its functioning is reflected by the cardiovascular variabilities and their relationships with the other physiological signals, especially the respiratory activity. Today LARY_CR is used only internally, in the framework of our clinical collaborations.
5. Software

5.1. Introduction

In order to validate our research results, our research activities encompass the development of related prototypes as surveyed below.

5.2. Emergent Middleware Enablers

Participant: Valérie Issarny [correspondent].

As part of our research work on Emergent Middleware, we have implemented Enablers (or Enabler functionalities) that make part of the overall CONNECT architecture realizing Emergent Middleware in practice.

Discovery Enabler: The CONNECT Discovery Enabler is the component of the overall CONNECT architecture that handles discovery of networked systems (NSS), stores their descriptions (NS models), and performs an initial phase of matchmaking to determine which pairs of systems are likely to be able to interoperate. Such pairs are then passed to the Synthesis Enabler so that mediators can be generated. The Discovery Enabler is written in Java and implements several legacy discovery protocols including DPWS and UPnP. Matchmaking is done on the basis of affordances contained in the system description, that is, ontological concepts describing the system’s category. Systems with the same affordance, or affordances standing in a specialization relationship, can be considered for connection. If a system does not provide its affordance, the Discovery Enabler can infer a likely one using text categorization based on the system’s interface description. The Discovery Enabler will soon be available for download under an open source license.

Synthesis Enabler: We have implemented (in Java) two approaches to mediator synthesis as part of the CONNECT Synthesis Enabler:

- **Mapping-based mediator synthesis.** This implementation focuses on networked systems that have compatible functionalities but are unable to interact successfully due to mismatching interfaces and/or behaviors. The ontology used in our implementation is encoded so as to make the reasoning more efficient at runtime while considering both subsumption and the union of classes. Based on the interface mapping, a correct-by-construction mediator is generated. In our current implementation, we are leveraging the LTS (Labeled Transition System Analyser) model checker to generate the parallel composition of the mapping processes and verify that the overall system successfully terminates. In the near future, we will be incorporating our techniques so as to deal with ambiguous mappings, i.e., when an action from one system may semantically be mapped to different actions from the other system.

- **Goal-based abstract mediator synthesis.** This implementation considers the protocols of two networked systems and produces the mediator protocol that allow them to interact so as to satisfy user goals. More specifically, the alphabet of the two protocols are aligned using ontology matching. The aligned protocols as well as the user goal are encoded as a satisfiability problem. The Zot model checker solves this problem (if possible) and produces a possible feasible interaction satisfying user goals.

The synthesis enabler will soon be available for download under an open source license.

5.3. Service-oriented Middleware for Pervasive Computing

Participants: Nikolaos Georgantas [correspondent], Valérie Issarny [correspondent].
In the past years, we have built a strong foundation of service-oriented middleware to support the pervasive computing vision. This specifically takes the form of a family of middlewares, all of which have been released under the open source LGPL license:

- **WSAMI - A Middleware Based on Web Services for Ambient Intelligence**: WSAMI (Web Services for AMbient Intelligence) is based on the Web services architecture and allows for the deployment of services on wireless handheld devices like smartphones and PDAs.
  URL: [http://www-rocq.inria.fr/arles/download/ozone/index.htm](http://www-rocq.inria.fr/arles/download/ozone/index.htm)

- **Ariadne - A Protocol for Scalable Service Discovery in MANETs**: Ariadne enriches WSAMI with the Ariadne service discovery protocol, which has been designed to support decentralized Web service discovery in multi-hop mobile ad hoc networks (MANETs). Ariadne enables small and resource-constrained mobile devices to seek and find complementary, possibly mobile, Web services needed to complete specified tasks in MANETs, while minimizing the traffic generated and tolerating intermittent connectivity.

- **MUSDAC - A Middleware for Service Discovery and Access in Pervasive Networks**: The Multi-protocol Service Discovery and ACcess (MUSDAC) middleware platform enriches WSAMI so as to enable the discovery and access to services in the pervasive environment, which is viewed as a loose and dynamic composition of independent networks. MUSDAC manages the efficient dissemination of discovery requests between the different networks and relies on specific plug-ins to interact with the various middleware used by the networked services.

- **INMIDIO - An Interoperable Middleware for Ambient Intelligence**: INMIDIO (INteroperable MIddleware for service Discovery and service InteractiOn) dynamically resolves middleware mismatch. More particularly, INMIDIO identifies the interaction middleware and also the discovery protocols that execute on the network and translates the incoming/outgoing messages of one protocol into messages of another, target protocol.

- **COCOA - A Semantic Service Middleware**: COCOA is a comprehensive approach to semantic service description, discovery, composition, adaptation and execution, which enables the integration of heterogeneous services of the pervasive environment into complex user tasks based on their abstract specification. Using COCOA, abstract user tasks are realized by dynamically composing the capabilities of services that are currently available in the environment.
  URL: [http://gforge.inria.fr/projects/amigo/](http://gforge.inria.fr/projects/amigo/)

- **ubiSOAP - A Service Oriented Middleware for Seamless Networking**: ubiSOAP brings multi-radio, multi-network connectivity to services through a comprehensive layered architecture: (i) the multi-radio device management and networking layers together abstract multi-radio connectivity, selecting the optimal communication link to/from nodes, according to quality parameters; (ii) the communication layer allows for SOAP-based point-to-point and group-based interactions in the pervasive network; and (iii) the middleware services layer brings advanced distributed resource management functionalities customized for the pervasive networking environment.
  URL: [http://www.ist-plastic.org](http://www.ist-plastic.org)

### 5.4. Supporting Service Orchestrations over Heterogeneous Interaction Paradigms

**Participant**: Nikolaos Georgantas [correspondent].
Established architectural paradigms enabling open system integration, such as service oriented architecture (SOA) and enterprise service bus (ESB), have provided answers to the essential issue of interoperability in distributed systems. However, realizations of these architectural paradigms fall short when it comes to integrating systems featuring heterogeneous interaction paradigms, such as client/server (CS), publish/subscribe (PS) and tuple space (TS), due to the differing interaction semantics of the latter. Typical solutions constitute in wrapping any system behind RPC-based service interfaces, which results in partial loss of interaction semantics. This can cause suboptimal or even problematic system integration.

Aiming at enabling seamless integration of heterogeneous interaction paradigms, we introduce an interoperability solution based on abstraction and merging of their common high-level semantics, paying special attention to the preservation of semantics. To this end, we propose three abstract connector types for the CS, PS and TS interaction paradigms. We further introduce a higher-level generic application (GA) connector type, which provides an abstract union of the three models, thus preserving their interaction semantics. We express our connector types in terms of application programming interface (API) primitives and related interaction protocol semantics. We then apply our abstractions to rethink a typical SOA- and ESB-based orchestration of heterogeneous distributed systems. Our solution features:

- Extending the BPEL workflow language with GA API primitives in terms of extension activities enabled by the BPEL specification;
- Introducing XSLT transformation between the GA-extended BPEL and the standard BPEL, which consists of encapsulating GA primitives into standard BPEL primitives and enables conveying GA semantics on top of BPEL primitives and subsequently on top of the common bus protocol primitives;
- Providing Java code templates for systematic and highly facilitated building of ESB-embedded binding components;
- Proposing interface description languages in the form of XSDs for systems employing CS-, PS-, TS-, and GA-connectors; and
- Introducing XSLT transformations between native system interface descriptions and GA-based interface descriptions.

We have developed our solution on top of the PEtALS ESB, which provides inherent support for BPEL by embedding the EasyBPEL workflow engine. Our solution considerably facilitates the application developer in designing and executing heterogeneous orchestrations. Furthermore, it is highly extensible, enabling easy integration of support for new middleware platforms. To demonstrate the applicability of our approach, we have implemented an application workflow integrating a JMEDS DPWS Web Service (CS), a JMS system based on Apache ActiveMQ (PS), and a Jini JavaSpaces system (TS). Our software will soon be released under open source license.

5.5. Srijan: Data-driven Macroprogramming for Sensor Networks

Participant: Animesh Pathak [correspondent].

Macroprogramming is an application development technique for wireless sensor networks (WSNs) where the developer specifies the behavior of the system, as opposed to that of the constituent nodes. As part of our work in this domain, we are working on Srijan, a toolkit that enables application development for WSNs in a graphical manner using data-driven macroprogramming.

It can be used in various stages of application development, viz.,

1. Specification of application as a task graph,
2. Customization of the auto-generated source files with domain-specific imperative code,
3. Specification of the target system structure,
4. Compilation of the macroprogram into individual customized runtimes for each constituent node of the target system, and finally
5. Deployment of the auto generated node-level code in an over-the-air manner to the nodes in the target system.
The current implementation of Srijan targets both the Sun SPOT sensor nodes and larger nodes with J2SE. Most recently, Srijan also includes rudimentary support for incorporating Web services in the application being designed. The software is released under open source license, and available as an Eclipse plug-in at http://code.google.com/p/srijan-toolkit/.

5.6. Yarta: Middleware for supporting Mobile Social Applications

Participant: Animesh Pathak [correspondent].

With the increased prevalence of advanced mobile devices (the so-called “smart” phones), interest has grown in Mobile Social Ecosystems (MSEs), where users not only access traditional Web-based social networks using their mobile devices, but are also able to use the context information provided by these devices to further enrich their interactions. We are developing a middleware framework for managing mobile social ecosystems, having a multi-layer middleware consisting of modules, which will provide the needed functionalities, including:

- Extraction of social ties from context (both physical and virtual),
- Enforcement of access control to protect social data from arbitrary access,
- A rich set of MSE management functionalities, using which mobile social applications can be developed.

Our middleware adopts a graph-based model for representing social data, where nodes and arcs describe socially relevant entities and their connections. In particular, we exploit the Resource Description Framework (RDF), a basic Semantic Web standard language that allows representing and reasoning about social vocabulary, and creating an interconnected graph of socially relevant information from different sources.

The current implementation of the Yarta middleware targets both desktop/laptop nodes running Java 2 SE, as well as Android smart phones. The software is released under open source license at https://gforge.inria.fr/projects/yarta/.

5.7. iBICOOP: Mobile Data Management in Multi-* Networks

Participant: Valérie Issarny [correspondent].

Building on the lessons learned with the development of pervasive service oriented middleware and of applications using them, we have been developing the custom iBICOOP middleware. iBICOOP specifically aims at assisting the development of advanced mobile, collaborative application services by supporting interactions between mobile users. Target application services in particular include the U-EVENT suite of services for professional events.

Briefly, the iBICOOP middleware addresses the challenges of easily accessing content stored on mobile devices, and consistent data access across multiple mobile devices by targeting both fixed and mobile devices, leveraging their characteristics (e.g., always on and unlimited storage for home/enterprise servers, ad hoc communication link between mobile devices), and by leveraging the capabilities of all available networks (e.g., ad hoc networks, Internet, Telecoms infrastructure networks). It also relies on Web and Telecoms standards to promote interoperability.

The base architecture of the iBICOOP middleware consists of core modules on top of which we can develop applications that may arise in the up-coming multi-device, multi-user world:

- The Communication Manager provides mechanisms to communicate over different available network interfaces of a device — Bluetooth, WiFi, Cellular — and also using different technologies e.g., Web services, HTTP/TCP sockets, ad hoc mode.
- The Security Manager uses well-established techniques of cryptography and secure communication to provide necessary security.
- The Partnership Manager provides device or user information in the form of profiles.
iBICOOP relies on service location protocols for naming and discovery of nearby services on currently active network interfaces that support IP multicast.

Besides normal file managing tasks, the Local File Manager gives the user clear cues to the files that have been replicated across multiple devices or shared among different users by using different icons.

The iBICOOP middleware has been licensed by AMBIENTIC (http://www.ambientic.com/), a start-up that specifically develops innovative mobile distributed services on top of the iBICOOP middleware that allows for seamless interaction and content sharing in today’s multi-* networks.

GANG Project-Team (section vide)
5. Software

5.1. RPL P2P

Participants: Emmanuel Baccelli [correspondant], Matthias Philipp.

P2P-RPL is an implementation of draft-ietf-roll-p2p-rpl, providing reactive discovery of point-to-point routes in low power and lossy networks such as wireless sensor networks. The implementation is based on the Contiki operating system. See also the web page http://contiki-p2p-rpl.gforge.inria.fr/.

- Version: 0.4

5.2. OPERA infrastructure

Participants: Cédric Adjih [correspondant], Ichra Amdouni, Pascale Minet, Ridha Soua.

OPERA-infrastructure is the system support code of OPERA, the Optimized Protocol for Energy efficient Routing with node Activity scheduling.

5.3. OPERA perf simul

Participants: Cédric Adjih [correspondant], Ichra Amdouni.

OPERA-perf-simul is a set of tools for simulation and performance evaluation as well as large scale tests of OPERA, the Optimized Protocol for Energy efficient Routing with node Activity scheduling.

5.4. OPERA protocol

Participants: Cédric Adjih [correspondant], Ichra Amdouni, Pascale Minet, Saoucene Mahfoudh.

OPERA-protocol is the heart of OPERA, the Optimized Protocol for Energy efficient Routing with node Activity scheduling. It includes EOND a neighborhood discovery protocol, EOSTC a protocol byuiding and maintaining a n energy efficient routing tree and SERENA a node coloring algorithm.

5.5. OPERA validation and tools

Participant: Cédric Adjih [correspondant].

OPERA-validation and tools is a set of tools for validation, debugging, analysis and visualization of OPERA protocol, the Optimized Protocol for Energy efficient Routing with node Activity scheduling. It operates either in a real embedded system or in simulation.
RAP Project-Team (section vide)
5. Software

5.1. Coccinelle

Participants: Julia Lawall [correspondent], Gilles Muller [correspondent], Gaël Thomas, Suman Saha, Arie Middlekoop.

Coccinelle is a program matching and transformation engine which provides the language SmPL (Semantic Patch Language) for specifying desired matches and transformations in C code. Coccinelle was initially targeted towards performing collateral evolutions in Linux. Such evolutions comprise the changes that are needed in client code in response to evolutions in library APIs, and may include modifications such as renaming a function, adding a function argument whose value is somehow context-dependent, and reorganizing a data structure.

Beyond collateral evolutions, Coccinelle has been successfully used for finding and fixing bugs in systems code. One of the main recent results is an extensive study of bugs in Linux 2.6 [51] that has permitted us to demonstrate that the quality of code has been improving over the last six years, even though the code size has more than doubled.


5.2. Telex

Participants: Marc Shapiro [correspondent], Lamia Benmouffok, Pierre Sutra, Pierpaolo Cincilla.

Developing write-sharing applications is challenging. Developers must deal with difficult problems such as managing distributed state, disconnection, and conflicts. Telex is an application-independent platform to ease development and to provide guarantees. Telex is guided by application-provided parameters: actions (operations) and constraints (concurrency control statements). Telex takes care of replication and persistence, drives application progress, and ensures that replicas eventually agree on a correct, common state. Telex supports partial replication, i.e., sites only receive operations they are interested in. The main data structure of Telex is a large, replicated, highly dynamic graph; we discuss the engineering trade-offs for such a graph and our solutions. Our novel agreement protocol runs Telex ensures, in the background, that replicas converge to a safe state. We conducted an experimental evaluation of the Telex based on a cooperative calendar application and on benchmarks.

We report on application experience, building a collaborative application for model-oriented software engineering above Telex, in SAC 2011 [50]. Future work includes extending Telex to cloud computing, opportunistic mobile networks, and real-time collaboration, within several ANR projects: PROSE (Section 7.1.5), STREAMS (Section 7.1.4) and ConcoRDanT (Section 7.1.3).

The code is freely available on http://gforge.inria.fr/ under a BSD license.

5.3. Treedoc

Participants: Marc Shapiro [correspondent], Marek Zawirski.

A Commutative Replicated Data Type (CRDT) is one where all concurrent operations commute. The replicas of a CRDT converge automatically, without complex concurrency control. We designed and developed a novel CRDT design for cooperative text editing, called Treedoc. It is designed over a dense identifier space based on a binary trees. Treedoc also includes an innovative garbage collection algorithm based on tree rebalancing.

In the best case, Treedoc incurs no overhead with respect to a linear text buffer. The implementation has been validated with performance measurements, based on real traces of social text editing in Wikipedia and SVN.
Work in 2010 has focused on studying large-scale garbage collection for Treedoc, and design improvements. Future work includes engineering a large-scale collaborative Wiki, and studying CRDTs more generally. This is the subject the PROSE, STREAMS and ConcoRDanT ANR projects (Sections 7.1.5, 7.1.4 and 7.1.3 respectively).

The code is freely available on http://gforge.inria.fr/ under a BSD license.

5.4. VMKit and .Net runtimes for LLVM

Participants: Harris Bakiras, Bertil Folliot [correspondent], Julia Lawall, Jean-Pierre Lozi, Gaël Thomas [correspondent], Gilles Muller, Thomas Preud’homme.

Many systems research projects now target managed runtime environments (MRE) because they provide better productivity and safety compared to native environments. Still, developing and optimizing an MRE is a tedious task that requires many years of development. Although MREs share some common functionalities, such as a Just In Time Compiler or a Garbage Collector, this opportunity for sharing has not been yet exploited in implementing MREs. We are working on VMKit, a first attempt to build a common substrate that eases the development and experimentation of high-level MREs and systems mechanisms. VMKit has been successfully used to build two MREs, a Java Virtual Machine and a Common Language Runtime, as well as a new system mechanism that provides better security in the context of service-oriented architectures.

VMKit project is an implementation of a JVM and a CLI Virtual Machines (Microsoft .NET is an implementation of the CLI) using the LLVM compiler framework and the MMTk garbage collectors. The JVM, called J3, executes real-world applications such as Tomcat, Felix or Eclipse and the DaCapo benchmark. It uses the GNU Classpath project for the base classes. The CLI implementation, called N3, is in early stages but can execute simple applications and the “pnetmark” benchmark. It uses the pnetlib project or Mono as its core library. The VMKit VMs compare in performance with industrial and top open-source VMs on CPU-intensive applications. VMKit is publicly available under the LLVM license.

http://vmkit.llvm.org/
5. Software

5.1. Gibbs’ Sampler

Participant: Chung Shue Chen.

The work on the self optimization of cellular networks based on Gibbs’ sampler (see Section 6.1.1.3) carried out in the joint laboratory with Alcatel-Lucent, led to the development of a software prototype that was presented by C. S. Chen at the INRIA Alcatel-Lucent joint laboratory seminar in March 2010 and demonstrated at the Alcatel-Lucent Bell Labs Open Days in May 2010. It was also demonstrated in the LINCS opening ceremony in April 2011.

5.2. PSI2

Participant: Ana Bušić.

The work on perfect sampling (see Section 6.2.3) has been partially implemented in a software tool PSI2, in collaboration with MESCAL team [INRIA Grenoble - Rhône-Alpes]; https://gforge.inria.fr/projects/psi.
5. Software

5.1. Syntax

Participants: Pierre Boullier [correspondant], Sattisvar Tandabany, Benoît Sagot.

See also the web page http://syntax.gforge.inria.fr/.

The (currently beta) version 6.0 of the SYNTAX system (freely available on INRIA GForge) includes various deterministic and non-deterministic CFG parser generators. It includes in particular an efficient implementation of the Earley algorithm, with many original optimizations, that is used in several of Alpage’s NLP tools, including the pre-processing chain SXPipe and the LFG deep parser SXLFG. This implementation of the Earley algorithm has been recently extended to handle probabilistic CFG (PCFG), by taking into account probabilities both during parsing (beam) and after parsing ($n$-best computation). SYNTAX 6.0 also includes parsers for various contextual formalisms, including a parser for Range Concatenation Grammars (RCG) that can be used among others for TAG and MC-TAG parsing.

Direct NLP users of SYNTAX for NLP, outside Alpage, include Alexis Nasr (Marseilles) and other members of the SEQUOIA ANR project (see section 8.2.1), Owen Rambow and co-workers at Columbia University (New York), as well as (indirectly) all SXPipe and/or SXLFG users. The project-team VASY (INRIA Rhône-Alpes) is one of SYNTAX’ users for non-NLP applications.

5.2. System DyALog

Participant: Éric Villemonte de La Clergerie [maintainer].

DYALOG on INRIA GForge: http://dyalog.gforge.inria.fr/

DYALOG provides an environment to compile and execute grammars and logic programs. It is essentially based on the notion of tabulation, i.e. of sharing computations by tabulating traces of them. DYALOG is mainly used to build parsers for Natural Language Processing (NLP). It may nevertheless be used as a replacement for traditional PROLOG systems in the context of highly ambiguous applications where sub-computations can be shared.

The current release 1.13.0 of DYALOG is freely available by FTP under an open source license and runs on Linux platforms for x86 and architectures and on Mac OS intel (both 32 and 64bits architectures). A partial port for Windows Cygwin has been successful but has not yet been integrated and finalized.

The current release handles logic programs, DCGs (Definite Clause Grammars), FTAGs (Feature Tree Adjoining Grammars), FTIGs (Feature Tree Insertion Grammars) and XRCGs (Range Concatenation Grammars with logic arguments). Several extensions have been added to most of these formalisms such as intersection, Kleene star, and interleave operators. Typed Feature Structures (TFS) as well as finite domains may be used for writing more compact and declarative grammars [127].

C libraries can be used from within DYALOG to import APIs (mysql, libxml, sqlite, ...).

DYALOG is largely used within ALPAGE to build parsers but also derivative softwares, such as a compiler of Meta-Grammars (cf. 5.3). It has also been used for building a parser from a large coverage French TIG/TAG grammar derived from a Meta-Grammar. This parser has been used for the Parsing Evaluation campaign EASY, the two Passage campaigns (Dec. 2007 and Nov. 2009), cf. [125, 126], and very large amount of data (700 millions of words) in the SCRIBO project.

DYALOG is used at LORIA (Nancy), University of Coruña (Spain), Instit Gaspard Monge (Univ. Marne La Vallée), University of Nice, and a few other users.

DYALOG and other companion modules are available on INRIA GForge.
5.3. Tools and resources for Meta-Grammars

**Participant:** Éric Villemonte de La Clergerie [maintainer].

**MGCOMP, MGTOOLS, and FRMG on INRIA GForge:** [http://mgkit.gforge.inria.fr/](http://mgkit.gforge.inria.fr/)

DYALOG (cf. [5.2](#)) has been used to implement MGCOMP, Meta-Grammar compiler. Starting from an XML representation of a MG, MGCOMP produces an XML representation of its TAG expansion.

The current version **1.5.0** is freely available by FTP under an open source license. It is used within ALPAGE and (occasionally) at LORIA (Nancy) and at University of Pennsylvania.

The current version adds the notion of namespace, to get more compact and less error-prone meta-grammars. It also provides other extensions of the standard notion of Meta-Grammar in order to generate very compact TAG grammars. These extensions include the notion of **Guarded nodes**, i.e. nodes whose existence and non-existence depend on the truth value of a guard, and the use of the regular operators provided by DYALOG on nodes, namely disjunction, interleaving and Kleene star. The current release provides a dump/restore mechanism for faster compilations on incremental changes of a meta-grammars.

The current version of MGCOMP has been used to compile a wide coverage Meta-Grammar FRMG (version **2.0.1**) to get a grammar of around 200 TAG trees [129]. Without the use of guarded nodes and regular operators, this grammar would have more than several thousand trees and would be almost intractable. FRMG has been packaged and is freely available.

To ease the design of meta-grammars, a set of tools have been implemented, mostly by Éric de La Clergerie, and collected in MGTOOLS (version **2.2.2**). This package includes a converter from a compact format to a XML pivot format, an Emacs mode for the compact and XML formats, a graphical viewer interacting with Emacs and XSLT stylesheets to derive HTML views. A new version is under development to provide an even more compact syntax and some checking mechanisms to avoid frequent typo errors.

The various tools on Metagrammars are available on INRIA GForge.

5.4. The Bonsai PCFG-LA parser

**Participants:** Benoît Crabbé [correspondant], Marie Candito, Pascal Denis, Djamé Seddah.

**Web page:** [http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html](http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html)

Alpage has developed as support of the research papers [81], [70], [71], [12] a statistical parser for French, named Bonsai, trained on the French Treebank. This parser provides both a phrase structure and a projective dependency structure specified in [4] as output. This parser operates sequentially: (1) it first outputs a phrase structure analysis of sentences reusing the Berkeley implementation of a PCFG-LA trained on French by Alpage (2) it applies on the resulting phrase structure trees a process of conversion to dependency parses using a combination of heuristics and classifiers trained on the French treebank. The parser currently outputs several well known formats such as Penn treebank phrase structure trees, Xerox like triples and CONLL-like format for dependencies. The parsers also comes with basic preprocessing facilities allowing to perform elementary sentence segmentation and word tokenisation, allowing in theory to process unrestricted text. However it is believed to perform better on newspaper-like text. The parser is available under a GPL license.

5.5. The MICA parser

**Participants:** Benoît Sagot [correspondant], Marie Candito, Pierre Boullier, Djamé Seddah.


MICA (Marseille-INRIA-Columbia- AT&T) is a freely available dependency parser [61] currently trained on English and Arabic data, developed in collaboration with Owen Rambow and Daniel Bauer (Columbia University) and Srinivas Bangalore (AT&T). MICA has several key characteristics that make it appealing to researchers in NLP who need an off-the-shelf parser, based on Probabilistic Tree Insertion Grammars and on the SYNTAX system. MICA is fast (450 words per second plus 6 seconds initialization on a standard high-end machine) and has close to state-of-the-art performance (87.6% unlabeled dependency accuracy on the Penn Treebank).
MICA consists of two processes: the supertagger, which associates tags representing rich syntactic information with the input word sequence, and the actual parser, based on the INRIA SYNTAX system, which derives the syntactic structure from the n-best chosen supertags. Only the supertagger uses lexical information, the parser only sees the supertag hypotheses.

MICA returns \( n \)-best parses for arbitrary \( n \); parse trees are associated with probabilities. A packed forest can also be returned.

5.6. Alpage’s linguistic workbench, including SxPipe

**Participants:** Benoît Sagot [correspondent], Rosa Stern, Pierre Boullier, Éric Villemonte de La Clergerie.

*See also the web page [http://lingwb.gforge.inria.fr/](http://lingwb.gforge.inria.fr/).*

Alpage’s linguistic workbench is a set of packages for corpus processing and parsing. Among these packages, the SxPipe package is of a particular importance. SxPipe, now in version 2 \[109\] is a modular and customizable chain aimed to apply to raw corpora a cascade of surface processing steps. It is used

- as a preliminary step before Alpage’s parsers (FRMG, SxLFG);
- for surface processing (named entities recognition, text normalization...).

Developed for French and for other languages, SxPipe 2 includes, among others, various named entities recognition modules in raw text, a sentence segmenter and tokenizer, a spelling corrector and compound words recognizer, and an original context-free patterns recognizer, used by several specialized grammars (numbers, impersonal constructions, quotations...).

5.7. MElt

**Participants:** Pascal Denis [correspondent], Benoît Sagot.

MElt is a part-of-speech tagger, trained for French (on the French TreeBank and coupled with the Lefff), English \[89\], Spanish, Kurmanji Kurdish \[131\] and Persian \[56\], \[42\]. It is state-of-the-art for French. It is distributed freely as a part of the Alpage linguistic workbench.

5.8. The Alexina framework: the Lefff syntactic lexicon, the Aleda entity database and other Alexina resources

**Participants:** Benoît Sagot [correspondent], Laurence Danlos.

*See also the web page [http://gforge.inria.fr/projects/alexina/](http://gforge.inria.fr/projects/alexina/).*

Alexina is Alpage’s Alexina framework for the acquisition and modeling of morphological and syntactic lexical information. The first and most advanced lexical resource developed in this framework is the Lefff, a morphological and syntactic lexicon for French.

Historically, the Lefff 1 was a freely available French morphological lexicon for verbs that has been automatically extracted from a very large corpus. Since version 2, the Lefff covers all grammatical categories (not just verbs) and includes syntactic information (such as subcategorization frames); Alpage’s tools, including Alpage’s parsers, rely on the Lefff. The version 3 of the Lefff, which has been released in 2008, improves the linguistic relevance and the interoperability with other lexical models.

Other Alexina lexicons are under development, in particular for Spanish (the Leffe), Polish, Slovak, English, Galician, Persian, Kurdish.

Alexina also hosts Aleda \[124\], an large-scale entity database currently developed for French but under development for English, extracted automatically from Wikipedia and Geonames. It is used among others in the SxPipe processing chain and its NP named entity recognition, as well as in the NOMOS named entity linking system.
5.9. The free French wordnet WOLF

Participants: Benoît Sagot [correspondant], Marianna Apidianaki.

The WOLF (Wordnet Libre du Français) is a wordnet for French, i.e., a lexical semantic database. The development of WOLF started in 2008 [113],[8]. At this time, we focused on benefitting from available resources of three different types: general and domain-specific bilingual dictionaries, multilingual parallel corpora and Wiki resources (Wikipedia and Wiktionaries). This work was achieved in a large part in collaboration with Darja Fišer (University of Ljubljana, Slovenia), in parallel with the development of a free Slovene wordnet, sloWNet. Since 2008, work specific to each of both resources has been done [114], but since end-2010 the collaboration has been re-activated. This is due among others to the fact that the joint development of WOLF and sloWNet is one of the main objectives of the two-year PROTEUS bilateral PHC project co-headed by Benoît Sagot (2010-2011, see section 8.3.2). Moreover, the EDyLex project also contributed to funding the improvement of the WOLF, in particular through the work of Marianna Apidianaki.

The WOLF is freely available under the Cecill-C license. It has already been used in various experiments, within and outside Alpage.

5.10. Automatic construction of distributional thesauri

Participants: Enrique Henestroza Anguiano [correspondant], Pascal Denis.

FREDIST is a freely-available (LGPL license) Python package that implements methods for the automatic construction of distributional thesauri [31].

We have implemented the context relation approach to distributional similarity, with various context relation types and different options for weight and measure functions to calculate distributional similarity between words. Additionally, FREDIST is highly flexible, with parameters including: context relation type(s), weight function, measure function, term frequency thresholding, part-of-speech restrictions, filtering of numerical terms, etc.

Distributional thesauri for French are also available, one each for adjectives, adverbs, common nouns, and verbs. They have been constructed with FreDist and use the best settings obtained in an evaluation. We use the L’Est Republicain corpus (125 million words), Agence France-Presse newswire dispatches (125 million words) and a full dump of the French Wikipedia (200 million words), for a total of 450 million words of text.

5.11. Tools and resources for time processing

Participants: Laurence Danlos [correspondant], Pascal Denis, Philippe Muller.

APETITE provides a set of tools to handle ISO-TimeML annotations, predict temporal structures from timex/event mark-ups, and different ways of evaluating the results. It is licensed under the Cecill, a GPL-like license http://www.irit.fr/~Philippe.Muller/tools/apepite-0.7.tgz .

In parallel, Alpage developed the French TimeBank [22],[21], a freely-available corpus annotated with ISO-TimeML-compliant temporal information (dates, events and relations between events).

5.12. System EasyRef

Participant: Éric Villemonte de La Clergerie [maintainer].

PASSAGE action

A collaborative WEB service EASYREF has been developed, in the context of ANR action Passage, to handle syntactically annotated corpora. EASYREF may be used to view annotated corpus, in both EASY or PASSAGE formats. The annotations may be created and modified. Bug reports may be emitted. The annotations may be imported and exported. The system provides standard user right management. The interface has been designed with the objectives to be intuitive and to speed edition.
EASYREF relies on a Model View Controller design, implemented with the Perl Catalyst framework. It exploits WEB 2.0 technologies (i.e. AJAX and JavaScript).

Version 2 has been used by ELDA and LIMSI to annotate a new corpus of several thousands words for PASSAGE.

A preliminary version 3 has been developed by François Guérin and revised by Éric de La Clergerie, relying on Berkeley DB XML to handle very large annotated corpora and to provide a complete query language expanded as XQuery expressions. EASYREF is maintained under INRIA GForge.
4. Software

4.1. Introduction

From its creation, AxIS has proposed new methods, approaches and software validated experimentally on various applications: Data Mining, Web usage Mining, Information Retrieval, Activity Modeling. Some of our results are under process to be part of the FocusLab platform (CPER Télus 5.5.2) which is based on a Service oriented Architecture. The development process has started this year, finding ways to fund human resources. Such a platform aims the community of Living Labs domain.

4.2. Data Mining

4.2.1. Classification and Clustering Methods

Participants: Marc Csernel, Yves Lechevallier [co-correspondant], Brigitte Trousse [co-correspondant].

We developed and maintained a collection of clustering and classification software, written in C++ and/or Java:

Supervised methods

- a Java library (Somlib) that provides efficient implementations of several SOM (Self-Organizing Map) variants [87], [86], [106], [105], [110], especially those that can handle dissimilarity data (available on Inria’s Gforge server (public access) https://gforge.inria.fr/projects/somlib/, developed by AxIS Rocquencourt and Brieuc Conan-Guez from Université de Metz.
- a functional Multi-Layer Perceptron library, called FNET, that implements in C++ supervised classification of functional data [101], [104], [103], [102] (developed by AxIS Rocquencourt).

Unsupervised methods: partitioning methods

- two partitioning clustering methods on the dissimilarity tables issued from a collaboration between AxIS Rocquencourt team and Recife University, Brazil: CDIs and CCClust [111]. Both are written in C++ and use the “Symbolic Object Language” (SOL) developed for SODAS. And one partitioning method on interval data (Div).
- two standalone versions improved from SODAS modules, SCluster and DIVCLUS-T [84] (AxIS Rocquencourt).

Unsupervised methods: agglomerative methods

- a Java implementation of the 2-3 AHC (developed by AxIS Sophia Antipolis). The software is available as a Java applet which runs the hierarchies visualization toolbox called HCT for Hierarchical Clustering Toolbox (see [85]).

A Web interface developed in C++ and running on our Apache internal Web server is available for the following methods: SCluster, Div, yCDis, CCClust.

Previous versions of the above software have been integrated in the SODAS 2 Software [98] which was the result of the european project ASSO 5 (2001-2004). SODAS 2 softsodaslinkware supports the analysis Stof multidimensional complex data (numerical and non numerical) coming from databases mainly in statistical offices and administration using Symbolic Data Analysis [82]. This software is registered at APP. The latest executive version of the SODAS 2 software, with its user manual can be downloaded at http://www.info.fundp.ac.be/asso/sodaslink.htm [88], [112].

5 ASSO: Analysis System of Symbolic Official data
4.2.2. Extracting Sequential Patterns with Low Support

**Participant:** Brigitte Trousse [correspondant].

Two methods for extracting sequential patterns with low support have been developed by D. Tanasa in his thesis (see Chapter 3 in [108] for more details) in collaboration with F. Masseglia and B. Trousse:

- **Cluster & Divide** [108]
- **Divide & Discover** [13], [108].

4.2.3. Mining Data Streams

**Participants:** Brigitte Trousse [correspondante], Mohamed Gaieb.

In Marascu’s thesis (2009) [95], a collection of software have been developed for knowledge discovery and security in data streams. Three **clustering methods for mining sequential patterns (Java) in data streams** method have been developed in Java:

- **SMDS** compares the sequences to each other with a complexity of $O(n^2)$.
- **SCDS** is an improvement of SMDS, where the complexity is enhanced from $O(n^2)$ to $O(n.m)$ with $n$ the number of navigations and $m$ the number of clusters.
- **ICDS** is a modification of SCDS. The principle is to keep the clusters’ centroids from one batch to another.

Such methods take batches of data in the format “Client-Date-Item” and provide clusters of sequences and their centroids in the form of an approximate sequential pattern calculated with an alignment technique.

In 2010 the Java code of one method called SCDS has been integrated in the MIDAS demonstrator (cf. 6.2.1) and a C++ version has been implemented by F. Masseglia for the CRE contract with Orange Labs with the deliverability of a licence) with a visualisation module (in Java).

It has been tested on the following data:

- Orange mobile portal logs (100 million records, 3 months) in the context of Midas project (Java version) and the CRE (Orange C++ version)
- Inria Sophia Antipolis Web logs (4 million records, 1 year, Java version)
- Vehicle trajectories (Brinkhoff generator http://iapg.jade-hs.de/personen/brinkhoff/generator/ in the context of MIDAS project (Java version)

This year it has been integrated as a Web service (Java version) in the first version of FocusLab platform in the ELLIOT context (cf. 5.5.2): a demonstration was made on San Rafaelle Hospital media use case at the first ELLIOT review at Brussels (cf. 6.3.1.1).

4.3. Web Usage Mining

4.3.1. AWLH for Pre-processing Web Logs

**Participants:** Yves Lechevallier [co-correspondant], Brigitte Trousse [co-correspondant].

**AWLH** (AxIS Web Log Preprocessing and Data Stream extraction) for Web Usage Mining (WUM) is issued from AxISlogminer preprocessing software which implements the multi-site log preprocessing methodology developed by D. Tanasa in his thesis [16] for Web Usage Mining (WUM). In the context of the Eiffel project (2008-2009), we isolated and redesigned the core of AxISlogMiner preprocessing tool (we called it AWLH) composed of a set of tools for pre-processing web log files. AWLH can extract and structure log files from several Web servers using different input format. The web log files are cleaned as usually before to be used by data mining methods, as they contain many noisy entries (for example, robots bring a lot of noise in the analysis of user behaviour then it is important in this case to identify robot requests). The data are stored within a database whose model has been improved.
Now the current version of our Web log processing (Available on INRIA's gforge website with private access) offers:

- Processing of several log files from several servers,
- Support of several input formats (CLF, ECLF, IIS, custom, ...);
- Incremental pre-processing;
- Java API to help integration of AWLH in external application.

An additional tool has been developed for capturing user actions in real time based on an open source project called "OpenSymphony ClickStream". An extension version of AWLH called AWLH-Debate has been developed for recording and structuring data issued from annotated documents inside discussion forums.

4.3.2. ATWUEDA for Analysing Evolving Web Usage Data

Participants: Yves Lechevallier [correspondant], Brigitte Trousse, Mohamed Gaieb, Yves Lechevallier.

ATWUEDA for Web Usage Evolving Data Analysis [90] was developed by A. Da Silva in her thesis [89] under the supervision of Y. Lechevallier. This tool was developed in Java and uses the JRI library in order to allow the application of R which is a programming language and software environment for statistical computing http://www.r-project.org/ functions in the Java environment.

ATWUEDA is able to read data from a cross table in a MySQL database. It splits the data according to the user specifications (in logical or temporal windows) and then applies the approach proposed in the Da Silva’s thesis in order to detect changes in dynamic environment. The proposed approach characterizes the changes undergone by the usage groups (e.g. appearance, disappearance, fusion and split) at each timestamp. Graphics are generated for each analyzed window, exhibiting statistics that characterizes changing points over time.


This year we have demonstrated the efficiency of ATWUEDA [51] by applyng it on another real case study on condition monitoring data streams of an electric power plant provided by EDF (cf. section 5.5.1).

ATWUEDA is used by Telecom Paris Tech and EDF [51].

4.4. Information Retrieval

4.4.1. CBR*Tools for Managing and Reusing Past Experiences based on Historical Data

Participant: Brigitte Trousse [correspondant].

CBR*Tools [92], [93] is an object-oriented framework [94], [91] for Case-Based Reasoning which is specified with the UMT notation (Rational Rose) and written in Java. It offers a set of abstract classes to model the main concepts necessary to develop applications integrating case-based reasoning techniques: case, case base, index, measurements of similarity, reasoning control. It also offers a set of concrete classes which implements many traditional methods (closest neighbors indexing, Kd-tree indexing, neuronal approach based indexing, standards similarities measurements). CBR*Tools currently contains more than 240 classes divided in two main categories: the core package for basic functionality and the time package for the specific management of the behavioral situations. The programming of a new application is done by specialization of existing classes, objects aggregation or by using the parameters of the existing classes.

CBR*Tools addresses application fields where the re-use of cases indexed by behavioral situations is required. The CBR*Tools framework was evaluated via the design and the implementation of several applications such as Broadway-Web, Educaid, BeCKB, Broadway-Predict, e-behaviour and Be-TRIP.

CBR*Tools is concerned by two past contracts: EPIA and MobiVIP.

CBR*Tools will be available for research, teaching and academic purpose via the FocusLab platform. The user manual can be downloaded at the URL: http://www-sop.inria.fr/axis/cbrtools/manual/.
See also the web page http://www-sop.inria.fr/axis/cbrtools/manual/.

4.4.2. Broadway*Tools for Building Recommender Systems on the Web

Participant: Brigitte Trousse [correspondant].

Broadway*Tools is a toolbox supporting the creation of adaptive recommendation systems on the Web or in an Internet/Intranet information system. The toolbox offers different servers, including a server that computes recommendations based on the observation of the user sessions and on the re-use of user groups’ former sessions. A recommender system created with Broadway*tools observes navigations of various users and gather evaluations and annotations, to draw up a list of relevant recommendations (Web documents, keywords, etc).

Based on Jaczynski’s thesis [92], different recommender systems have been developed for supporting Web browsing, but also browsing inside a Web-based information system or for query formulation in the context of a meta search engine.

4.5. Activity Modeling

4.5.1. K-MADe for Describing Human Operator or User Activities

Participant: Dominique Scapin [correspondant].

K-MADe tool (Kernel of Model for Human Activity Description Environment). The K-MADe is intended for people wishing to describe, analyze and formalize the activities of human operators, of users, in environments (computerized or not), in real or simulated situation; in the field, or in the laboratory. Although all kinds of profiles of people are possible, this environment is particularly intended for ergonomics and HCI (Human Computer Interaction) specialists. It has been developed through collaboration between ENSMA (LISI XSlaboratory) and INRIA. The last release was delivered on November 1st 2010 based on the work of Caffiau and al. [83].

Its history, documentation and tool are available at: http://kmade.sourceforge.net/index.php
5. Software

5.1. ElevatorRoutePlanner
Participants: Fawzi Nashashibi [correspondant], Paulo Lopes Resende.

This software is dedicated to the building of a decision system that performs task planning and especially route planning for a multiple vehicles based system. Each vehicle sends remotely and asynchronously its position and speed to the system and the “elevator route planner” decides for the destination of each vehicle among a predefined map of fixed stations. Since the stations are on one side of the road, the vehicles are possibly sharing the same stations, leading sometimes to conflictual situations the system has to solve.

• Version: V1

5.2. MELOSYM
Participants: Fawzi Nashashibi [correspondant], Jianping Xie.

MELOSYM is the acronym for “Modélisation de l’Environnement et LOcalisation en temps réel pour un SYstème Mobile autonome ou pas, fondé sur des données du capteur laser”. This is a SLAM based algorithm for the environment mapping and vehicle localisation in real time using laser data. The particularity of the algorithm is its hierarchical approach that improves the accuracy of the system and speeds up the computations.

• Version: V1

5.3. ObstaclesDetectionLaser
Participants: Fawzi Nashashibi [correspondant], Paulo Lopes Resende, Laurent Bouraoui.

This is a software for obstacle detection by processing laser range finders data and return the position and distance of the nearest obstacle with regard to the vehicle. The data are from single or multi-layer sensors with different scopes and ranges. The sensors can operate simultaneously or individually, synchronously or not.

• Version: v1

5.4. Path2TrajectoryPlanner
Participants: Fawzi Nashashibi [correspondant], Paulo Lopes Resende.

This software can calculate the exact trajectory of a vehicle from its route decided by another decision-making system. The trajectory is expressed in terms of position and orientation and velocity versus time.

• Version: V1

5.5. SimpleController
Participants: Fawzi Nashashibi [correspondant], Paulo Lopes Resende.

This software enables the development of simple commands or controllers to be applied to drive members (actuators) of a vehicle allowing it to perform a pre-calculated trajectory. The component implements a path following controller. It takes as inputs a trajectory and a vehicle state and it determines the steering and velocity command to be performed by a car type vehicle.

• Version: V1
5.6. CCI

**Participants:** Fawzi Nashashibi [correspondant], Carlos Holguin.

This software provides a visual and audio interface for the users in a vehicle, in which they can select a destination and see the status of the trip and vehicle systems. It is formed by a component that runs in RTMaps that communicates through a tunnel with a C# application.

- Version: V1
5. Software

5.1. IKONA/MAESTRO Software

Participants: Vera Bakić, Nozha Boujemaa, Jean-Paul Chièze, Raffi Enficiaud, Alexis Joly, Laurent Joyeux, Olfa Mzoughi, Souheil Selmi, Itheri Yahiaoui.

IKONA is a generalist software dedicated to content-based visual information indexing and retrieval. It has been designed and implemented in our team during the last years [21]. Its main functionalities are the extraction, the management and the indexing of many state-of-the-art global and local visual features. It offers a wide range of interactive search and navigation methods including query-by-example, query-by-window, matching, relevance feedback, search results clustering or automatic annotation. It can manage several types of input data including images, videos and 3D models.

Based on a client/server architecture, it is easily deployable in any multimedia search engine or service. The communication between the two components is achieved through a proprietary network protocol. It is a set of commands the server understands and a set of answers it returns to the client. The communication protocol is extensible, i.e. it is easy to add new functionalities without disturbing the overall architecture. It can be replaced by any new or existing protocol dealing with multimedia information retrieval.

The main processes are on the server side. They can be separated in two main categories:

- off-line processes: data analysis, features extraction and structuration
- on-line processes: answer the client requests

Several clients can communicate with the server. A good starting point for exploring the possibilities offered by IKONA is our web demo, available at http://www-roc.inria.fr/cgi-bin/imedia/circario.cgi/bio_diversity?select_db=1. This CGI client is connected to a running server with several generalist and specific image databases, including more than 23,000 images. It features query by example searches, switch database functionality and relevance feedback for image category searches. The second client is a desktop application. It offers more functionalities. More screen-shots describing the visual searching capabilities of IKONA are available at http://www-rocq.inria.fr/imedia/cbir-demo.html.

IKONA is a pre-industrial prototype, with exploitation as a final objective. Currently, there does not exist a licensed competitor with the same range of functionalities. It exists several commercial softwares or systems exploiting technologies similar to some functionalities of IKONA but usually not the most advanced ones. We can for example cite the SDK developed by LTU company, the service proposed by AdVestigo company, etc. Many prototypes and demonstrators, industrial or academic, share some functionalities of IKONA but here again not the most advanced (e.g. Google Image Similarity Search Beta, IBM Muffin, etc.).

The main originality of IKONA is its genericity (in terms of visual features, metrics, input data, storage format, etc.), its adaptivity (to new visual features, new indexing structures or new search algorithms), its innovative interactive search functionalities (Local and Global Relevance Feedback, Local Search with Query Expansion, Search results clustering, etc.) and its scalability thanks to a generic indexing structure module than can support the integration of any new advances.

Current Users of IKONA include European and National Projects Participants through its integration in prototype multimedia systems, commercial companies through user trials (EXALEAD, INA, BELGA, AFP), General or Specific Public through Web demos (Pl@ntNet leaf identification demo).

IKONA software provides a high degree of visibility for IMEDIA scientific works through demos in commercial, scientific and general public events (notably in most INRIA national showrooms). It is also the mainstay of several Multimedia Systems developed at the European level, in conjunction with many Leader European Companies and Research Centers.
5. Software

5.1. Introduction

In our research domain, developing software prototypes is mandatory to validate research solutions and is an important vector for publications, demonstrations at conferences and exhibitions as well as for cooperations with industry. This prototyping task is however difficult because it requires specialized hardware platforms (e.g., new generations of smart tokens), themselves sometimes at an early stage of development.

For a decade, we have developed successive prototypes addressing different application domains, introducing different technical challenges and relying on different hardware platforms. PicoDBMS was our first attempt to design a full-fledged DBMS embedded in a smart card [9] [27]. Chip-Secured Data Access (C-SDA) embedded a reduced SQL query engine and access right controller in a secure chip and acted as an incorruptible mediator between a client and an untrusted server hosting encrypted data [33]. Chip-Secured XML Access (C-SXA) was an XML-based access rights controller embedded in a smart card [8]. Prototypes of C-SXA have been the recipient of the e-gate open 2004 Silver Award and SIMagine 2005 Gold award, two renowned international software contests. The next subsections details the two prototypes we are focusing on today.

5.2. PlugDB engine

Participant: Nicolas Anciaux.

More than a stand-alone prototype, PlugDB is part of a complete architecture dedicated to a secure and ubiquitous management of personal data. PlugDB aims at providing an alternative to a systematic centralization of personal data. To meet this objective, the PlugDB architecture lies on a new kind of hardware device called Secure Portable Token (SPT). Roughly speaking, a SPT combines a secure microcontroller (similar to a smart card chip) with a large external Flash memory (Gigabyte sized). The SPT can host data on Flash (e.g., a personal folder) and safely run code embedded in the secure microcontroller. PlugDB engine is the cornerstone of this embedded code. PlugDB engine manages the database on Flash (tackling the peculiarities of NAND Flash storage), enforces the access control policy defined on this database, protects the data at rest against piracy and tampering, executes queries (tackling low RAM constraint) and ensures transaction atomicity. Part of the on-board data can be replicated on a server (then synchronized) and shared among a restricted circle of trusted parties through crypto-protected interactions. PlugDB engine has been registered at APP (Agence de Protection des Programmes) in 2009 [29] and its Flash-based indexing system has been patented by INRIA and Gemalto [37]. It has been demonstrated in a dozen of national and international events including JavaOne and SIGMOD. It is being experimented in the field to implement a secure and portable medical-social folder helping the coordination of medical care and social services provided at home to dependent people.


5.3. uFLIP Benchmark

Participant: Luc Bouganim.

It is amazingly easy to produce meaningless results when measuring flash devices, partly because of the peculiarity of flash memory, but primarily because their behavior is determined by layers of complex, proprietary, and undocumented software and hardware. uFLIP is a component benchmark for measuring the response time distribution of flash IO patterns, defined as the distribution of IOs in space and time. uFLIP includes a benchmarking methodology which takes into account the particular characteristics of flash devices. The source code of uFLIP, available on the web (700 downloads, 4000 distinct visitors), was registered at APP in 2009 [31]. It has been demonstrated at SIGMOD.

WILLOW Project-Team

5. Software

5.1. SPArse Modeling Software (SPAMS)

SPAMS v2.1 was released as open-source software in June 2011 (v1.0 was released in September 2009 and v2.0 in November 2010). It is an optimization toolbox implementing algorithms to address various machine learning and signal processing problems involving

- Dictionary learning and matrix factorization (NMF, sparse PCA, ...)
- Solving sparse decomposition problems with LARS, coordinate descent, OMP, SOMP, proximal methods
- Solving structured sparse decomposition problems ($\ell_1/\ell_2$, $\ell_1/\ell_\infty$, sparse group lasso, tree-structured regularization, structured sparsity with overlapping groups,...).

The software and its documentation are available at http://www.di.ens.fr/willow/SPAMS/.

5.2. Non-uniform Deblurring for Shaken and Partially Saturated Images

This is a package of Matlab code for non-blind removal of non-uniform camera shake blur from a single blurry image. The package explicitly deals with images containing some saturated pixels. The algorithm is described in [19]. The package is publicly available at http://www.di.ens.fr/willow/research/saturation/.

5.3. Local dense and sparse space-time features

This is a package with Linux binaries implementing extraction of local space-time features in video. The package was updated in January 2011. The code supports feature extraction at Harris3D points, on a dense space-time grid as well as at user-supplied space-time locations. The package is publicly available at http://www.di.ens.fr/~laptev/download/stip-2.0-linux.zip.

5.4. Segmenting Scenes by Matching Image Composites

This is a package of Matlab code implementing unsupervised data-driven scene segmentation as described in (Russell et al. NIPS 2009). The package was created in June 2011 and is available at http://www.cs.washington.edu/homes/bcr/projects/SceneComposites/index.html.

5.5. Discriminative Clustering for Image Co-segmentation

This is a package of Matlab code implementing unsupervised discriminative clustering for co-segmenting multiple images described in (Joulin et al. CVPR 2010) and (Joulin et al. NIPS 2010). The aim is to segment a given set of images containing objects from the same category, simultaneously and without prior information. The package was last updated in October 2011 and is available at http://www.di.ens.fr/~joulin/code/coseg.zip.

5.6. Clustering with Convex Fusion Penalties

This is a package of Matlab code implementing a hierarchical clustering with convex fusion penalties described in (Hocking et al. ICML 2011 [10]). The package is available at http://www.di.ens.fr/~joulin/code/clusterpath_norm_Inf.zip.