RESEARCH CENTER

FIELD
Networks, Systems and Services, Distributed Computing

Activity Report 2012

Section Software

Edition: 2013-04-24

ACES Project-Team (section vide)

ADAM Project-Team

5. Software

5.1. Introduction

We report on the major software systems that are developed by our research group. FRASCATI, PowerAPI, SPACES relate to the first research direction. AppliIDE and CALICO relate to the second one. Finally, FRACTAL is a general purpose component framework that serves as a foundation for most of our work around reconfigurable middleware.

5.2. AppliIDE

Participants: Laurence Duchien, Clément Quinton [correspondant].

AppliIDE is directly connected to the work of Carlos Parra's PhD thesis [116] and Ubinov ADT's work which covers the definition and implementation of a Context-Aware Dynamic Software Product Line (DSPL) named CAPucine. It provides a set of tools for the selection of features, metamodel transformation and code generation for mobile applications [119]. The current implementation of ApplIDE addresses transformation from CAPucine metamodel towards SCA metamodel, and Spoon EMF metamodel. The transformations were formerly written with Acceleo tool, which is a dedicated language for transformation, enhancing the readability. ApplIDE metamodels are based on the Eclipse Modeling Framework. Code generators are all written in Acceleo.

Inria Evaluation Committee Criteria for Software Self-Assessment: A-3, SO-3, SM-3, EM-2, SDL-2. ApplIDE is registered with the APP (Agence pour la Protection des Programmes) under reference IDDN.FR.001.500004.000.S.A.2010.000.10600.

5.3. CALICO

Participants: Laurence Duchien [correspondant], Antonio de Almeida Souza Neto.

CALICO is an agile development framework for the design and evolution of safe component-based and service-oriented software that has been developed in the context of Guillaume Waignier's PhD thesis [128].

Agile software development relies on an iterative and incremental development cycle that allows the software architect to iterate between the design of the architecture and the debug of the software in its execution context. At each iteration, the architect can evolve its software and check the consistency of its evolution through the execution of static and dynamic analysis tools. During the design and the evolution of the system, the architect can use a set of metamodels to specify the structure of the architecture and its various quality of services requirement. During the deployment, CALICO instantiates the system on the target runtime platform from the models specified and keeps them synchronized with the software during its execution. Through this means, the architect has a conceptual view, which allows him to reason on the critical software properties during its evolution. Moreover, in order to check these evolutions, CALICO provides a unifying framework, which allows reuse of many static analysis tools of software architectures and dynamic debugging tools, that were scattered in different existing platforms. Thus, each change can be statically analyzed on the conceptual view before being propagated to the software system. Dynamic analyses are based on data values available during the execution only. The capture of these values is done through automatic instrumentation of the software system.

Globally, CALICO enables reliable evolution even if the underlying platforms do not natively provide this support. The current version handles four component-based and service-oriented platforms (FRASCATI, FRACTAL, OPENCCM, OPENCOM). Moreover, the benchmarks that we have performed show that CALICO is usable for the design and development of safe applications up to 10,000 components and services, which corresponds to the maximal load of most runtime platforms.

Inria Evaluation Committee Criteria for Software Self-Assessment: A-3, SO-4, SM-3, EM-3, SDL-4. CALICO is an open source software available at http://calico.gforge.inria.fr.

5.4. Fractal

Participants: Philippe Merle [correspondant], Romain Rouvoy, Lionel Seinturier.

FRACTAL is a modular, extensible, and reflective component framework. The FRACTAL toolchain can be used to design, implement, deploy and reconfigure any kind of software and middleware system. FRACTAL has initially been designed by both Inria and France Telecom R&D.

Inria Evaluation Committee Criteria for Software Self-Assessment: A-4, SO-4, SM-3-up, EM-3-up, SDL-4-up, DA-3, CD-4, MS-4, TPM-4. FRACTAL is a project of the OW2 consortium for open-source middleware. Web site: http://fractal.ow2.org. License LGPL. Some of the research activities around FRACTAL [91], [90], [124] are on top cited publications of the CBSE research community [109]. The ADAM project-team members are among the top committers of the project with 33.8% of all commits and they are the principal contributors for several modules including AOKell [124], Fraclet, the Inria ODL F4E [95], [96], Juliac, Koch. Philippe Merle is the leader of the OW2 FRACTAL project.

5.5. FraSCAti

Participants: Philippe Merle [correspondant], Christophe Munilla, Romain Rouvoy, Lionel Seinturier.

FRASCATI is a service-oriented component-based middleware platform implementing OASIS *Service Component Architecture* (SCA) specifications.The main originality of OW2 FRASCATI is to bring FRACTAL-based reflectivity to SCA, *i.e.*, any FRASCATI software component is equipped with both the SOA capabilities brought by SCA and the reflective capabilities (*i.e.*, introspection and reconfiguration) brought by FRACTAL. Various micro-benchmarks have shown that FRASCATI reflectivity is achieved without hindering its performance relative to the de facto reference SCA implementation, *i.e.*, Apache Tuscany. Non-functional concerns (logging, transaction, security, etc.), so called intents in SCA terms, are also programmed as FRASCATI components and are (un)woven on business components dynamically at runtime, this is based on aspect-oriented concepts defined in FAC [117]. OW2 FRASCATI supports various implementation technologies (SCA Composite, Java, WS-BPEL, Spring Framework, OSGi, Fractal ADL, native C library, Apache Velocity templates, and seven scripting languages as BeanShell, FScript, Groovy, JavaScript, JRuby, Jython, XQuery) for programming services or integrating legacy code, various binding protocols (SOAP, REST, JSON-RPC, UPnP, HTTP servlets, Java RMI, JMS, JGroups) and interface definition languages (WSDL, Java, WADL) for interoperating with existing services. OW2 FRASCATI provides management tools like standalone, Web-based, and JMX-based graphical consoles and a dedicated scripting language for reconfiguring SCA applications. The whole OW2 FRASCATI platform is itself built as a set of reflective SCA components.

Inria Evaluation Committee Criteria for Software Self-Assessment: A-4-up, SO-4, SM-4-up, EM-3-up, SDL-4-up, DA-4, CD-4, MS-4, TPM-4. FRASCATI is a project of the OW2 consortium for open-source middleware. Web site: http://frascati.ow2.org. 208 Kloc (mainly Java). Registered with the APP (Agence pour la Protection des Programmes) under reference FR.001.050017.000.S.P.2010.000.10000. License: LGPL. Embedded into several industrial software systems: EasySOA, Petals Link EasyViper, EasyBPEL, EasyESB, OW2 PEtALS, OW2 Scarbo. Various demonstrators built during funded projects: ANR SCOrWare, FP7 SOA4All, ANR ITEmIS, ANR SALTY, ANR SocEDA, FUI Macchiato, FUI EasySOA, ADT Galaxy and ADT Adapt. Main publications: [19], [123], [111], [112], [98], [97].

5.6. PowerAPI

Participants: Aurélien Bourdon, Adel Noureddine, Romain Rouvoy [correspondant].

PowerAPI is a Scala-based library for monitoring energy at the process-level. It is based on a modular and asynchronous event-driven architecture using the Akka library. PowerAPI differs from existing energy process-level monitoring tool in its pure software, fully customizable and modular aspect which let users precisely define what they want to monitor, without plugging any external device. PowerAPI offers an API which can be used to express requests about energy spent by a process, following its hardware resource utilization (in terms of CPU, memory, disk, network, etc.). Its applications cover energy-driven benchmarking [52], energy hotspots and bugs detection [53], [75] and real-time distributed system monitoring.

PowerAPI is registered with the APP (Agence pour la Protection des Programmes) under reference IDDN.FR.001.400015.000.S.P.2012.000.10000. License: AGPL.

5.7. SPACES

Participants: Russel Nzekwa, Daniel Romero [correspondant], Romain Rouvoy, Lionel Seinturier.

SPACES is a context mediation middleware that follows the *REpresentational State Transfer* (REST) principles [100]. The current implementation of SPACES is based on the COSMOS context framework [93], [121] and the COMANCHE web server [91]. Both COSMOS and COMANCHE are based on the FRACTAL component model and the JULIA implementation [91].

The main features of the current SPACES implementation are presented below:

1. *Ubiquitous connectors*: SPACES defines connectors that encapsulate the distribution concern. These connectors expose the COSMOS context nodes as REST resources with logical associated URLs, and enable interactions between consumers and producers via different communication protocols and the discovery of the available context sources. The current SPACES implementation supports interaction using the HTTP and twitter [108] protocols. For discovery, the implementation uses the Service Location Protocol (SLP) [102].
2. *Context Representation*: Following the REST principles, SPACES supports multiple representations of the context information: JSON [94], XML and Java serialization.
3. *Quality of context (QoC) information*: The QoC properties are incorporated as service attributes in the SLP advertisements of the context information.
4. *Context selection*: The restrictions in terms of QoC of the required context information are expressed as LDAP filters [125]. SPACES benefits from the LDAP based queries of SLP to select the context providers.

We use XStream 1.3.13 [89] and JSON-lib 2.2.34 [88] to serialize context information as XML and JSON documents. For SLP and twitter we employ jSLP 1.0.0 [120] and twitter-4j 2.0.6 [129].

SPACES is registered with the APP (Agence pour la Protection des Programmes) under reference IDDN 10-500002-000.

ARLES Project-Team

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. iCONNECT – 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 [2]. The focus of ARLES work is on the: *Discovery enabler* that builds on our extensive background in the area of interoperable pervasive service discovery; and *Synthesis enabler* that synthesizes mediators that allow networked systems that have compatible functionalities to interact despite mismatching interfaces and/or behaviors.

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.

The proposed solution to mediator synthesis assumes an ontology-based system description *à la* OWL-S, which is made available by the Discovery Enabler, possibly helped by machine learning. The semantically-annotated interfaces of systems that need to communicate are then processed to compute the semantic mapping between their respective operations using a constraint solver. The resulting mapping serves generating a mediator process that further coordinates the behaviors of the systems and guarantees their successful interaction. The mediator is deployed on a dedicated engine able to parse and compose middleware messages and convert them to fit each system expectations regarding the type and order of these messages.

The CONNECT Enablers have been integrated and experimented with by the CONNECT consortium to effectively enable Emergent Middleware. Part of them are available for download under an open source license at the CONNECT Web site at https://www.connect-forever.eu/software.html.

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
- **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.

URL: http://www-rocq.inria.fr/arles/download/ariadne/index.html

- **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.
  URL: http://www-rocq.inria.fr/arles/download/ubisec/index.html
- **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.
  URL: http://www-rocq.inria.fr/arles/download/inmidio/index.html
- **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/
- **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.

5.4. xSOM – Service-oriented middleware for the Future Internet

Participant: Nikolaos Georgantas [correspondent].

Building on our long experience on service-oriented middleware (SOM) for pervasive environments (see § 5.3 above) and given the evolution of such environments towards the Future Internet and the Internet of Things, we have already implemented early results of our related research into an extensible SOM (xSOM) for the Future Internet. xSOM aims at enabling large-scale dynamic compositions of services and things, while being highly extensible for accommodating the extreme heterogeneity of such compositions. xSOM currently supports two major functionalities: (i) a protocol bus-based solution to seamless integration of heterogeneous interaction paradigms for services and things; and (ii) a solution to discovery, access and data fusion over large populations of things.

Regarding (i), we have introduced a protocol interoperability solution comprising representative abstract connector types for the client/server (CS), publish/subscribe (PS) and tuple space (TS) paradigms, as well as their mapping to a higher-level generic application (GA) connector type. We apply these connector abstractions to introduce an enhanced bus paradigm, the eXtensible Service Bus (XSB). XSB features richer interaction semantics than common Enterprise Service Bus (ESB) implementations and incorporates special consideration for semantics-preserving cross-integration of CS, PS and TS. We have carried out a realization of XSB —first on the PEtALS ESB, and then on EasyESB— where we provide templates for systematic and highly facilitated building of binding components for heterogeneous systems (services and things) that are plugged into the XSB. To demonstrate the applicability of our approach, we have implemented a smoke-detection-and-alert system integrating a JMEDS DPWS Web Service (CS), a JMS system based on Apache ActiveMQ (PS), and a Jini JavaSpaces system (TS).

Regarding (ii), we support data queries over large populations of things, notably smartphones, which are getting increasingly ubiquitous and embed a rich collection of sensors. xSOM enables: (i) high-level programming of things on top of heterogeneous smartphone sensors; (ii) thing discovery and access dealing with large numbers of networked things; and (iii) on-the-fly composition of such things and fusion of their data in response to queries. In target settings, e.g., at the scale of a city, not all phones need to register for reporting their data (e.g., ambient sound level); some smartly distributed sampling is sufficient. This enables efficient scalable coverage of the entire city with only a subset of the large phone population being registered. The phone's things registration manager includes a probabilistic decision algorithm for selective registration based on the truncated Lévy walk mobility model. The registration decision is based on the actual density of already registered phones, the coverage quality requirements, and the coverage of the estimated path that the user will take for the next few minutes. We have implemented a demonstrative application enabling a user to know "how lively is this city spot at this moment' based on retrieving and aggregating smartphone ambient sound level data.

Our software will soon be released under open source license as part of the newly launched OW2 initiative on "Future Internet of Software Services" (http://www.ow2.org/view/Future_Internet/).

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 architecture 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.

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.
URL: https://www-roc.inria.fr/arles/index.php/ongoing-research-projects/74-data-sharing-and-replication-in-pervasive-networks

ASAP Project-Team

5. Software

5.1. WhatsUp: A Distributed News Recommender

Participants: Antoine Boutet, Davide Frey, Arnaud Jegou, Anne-Marie Kermarrec.

Contact:	Antoine Boutet
Licence:	Open Source
Presentation:	A Distributed News Recommender
Status:	Beta version

This work has lead to the development of WhatsUp, a distributed recommendation system aimed to distribute instant news in a large scale dynamic system. WhatsUp has two parts, an embedded application server in order to exchange with other peers in the system and a fully dynamic web interface for displaying news and collecting opinions about what the user reads. Underlying this web-based application lies Beep, a biased epidemic dissemination protocol that delivers news to interested users in a fast manner while limiting spam. Beep is parametrized on the fly to manage the orientation and the amplification of news dissemination. Every user forwards the news of interest to a randomly selected set of users with a preference towards those that have similar interests (orientation). The notion of interest does not rely on any explicit social network or subscription scheme, but rather on an implicit and dynamic overlay capturing the commonalities between users with respect to what they are interested in. The size of the set of users to which a news is forwarded depends on the interest of the news (amplification). A centralized version of WhatsUp is already up and running and the decentralized one is still in beta version.

5.2. GossipLib: effective development of gossip-based applications

Participants: Davide Frey, Heverson Borba Ribeiro, Anne-Marie Kermarrec.

Contact:	Davide Frey
Licence:	Open Source
Presentation:	Library for Gossip protocols
Status:	released version 0.7alpha

GossipLib is a library consisting of a set of JAVA classes aimed to facilitate the development of gossip-based application in a large-scale setting. It provides developers with a set of support classes that constitute a solid starting point for building any gossip-based application. GossipLib is designed to facilitate code reuse and testing of distributed applications: it thus provides the implementation of a number of standard gossip protocols that may be used out of the box or extended to build more complex protocols and applications. These include for example the peer-sampling protocols for overlay management.

GossipLib also provides facility for the configuration and deployment of applications as final-product but also as research prototype in environments like PlanetLab, clusters, network emulators, and even as event-based simulation. The code developed with GossipLib can be run both as a real application and in simulation simply by changing one line in a configuration file.

5.3. YALPS

Participants: Davide Frey, Heverson Borba Ribeiro, Anne-Marie Kermarrec.

Contact:	Davide Frey
Licence:	Open Source
Presentation:	Library for Gossip protocols
Status:	released version 0.3alpha

YALPS is an open-source Java library designed to facilitate the development, deployment, and testing of distributed applications. Applications written using YALPS can be run both in simulation and in real-world mode without changing a line of code or even recompiling the sources. A simple change in a configuration file will load the application in the proper environment. A number of features make YALPS useful both for the design and evaluation of research prototypes and for the development of applications to be released to the public. Specifically, YALPS makes it possible to run the same application as a simulation or in a real deployment without a single change in the code. Applications communicate by means of application-defined messages which are then routed either through UDP/TCP or through YALPS's simulation infrastructure. In both cases, YALPS's communication layer offers features for testing and evaluating distributed protocols and applications. Communication channels can be tuned to incorporate message losses or to constrain their outgoing bandwidth. Finally, YALPS includes facilities to support operation in the presence of NATs and firewalls using relaying and NAT-traversal techniques.

The work has been done in collaboration with Maxime Monod (EPFL).

5.4. HEAP: Heterogeneity-aware gossip protocol.

Participants: Davide Frey, Arnaud Jegou, Anne-Marie Kermarrec.

Contact:	Davide Frey
Licence:	Open Source
Presentation:	Java Application
Status:	release & ongoing development

This work has been done in collaboration with Vivien Quéma (CNRS Grenoble), Maxime Monod and Rachid Guerraoui (EPFL), and has lead to the development of a video streaming platform based on HEAP, *HEterogeneity-Aware gossip Protocol*. The platform is particularly suited for environment characterized by heterogeneous bandwidth capabilities such as those comprising ADSL edge nodes. HEAP is, in fact, able to dynamically leverage the most capable nodes and increase their contribution to the protocol, while decreasing by the same proportion that of less capable nodes. During the last few months, we have integrated HEAP with the ability to dynamically measure the available bandwidth of nodes, thereby making it independent of the input of the user.

ASCOLA Project-Team

5. Software

5.1. AWED

Participants: Mario Südholt [correspondent], Ismael Mejia.

Aspect-oriented programming, distributed programming, event-based programming, invasive patterns

The model of Aspects With Explicit Distribution (AWED) supports the modularization of crosscutting functionalities of distributed applications. It addresses the problem that common aspect systems do not provide features for distributed programming. It notably features three main aspect abstractions: remote pointcuts, remotely-executed advice, and distributed aspects.

The AWED system has also been employed in the CESSA project proposal (see Sec. 8.1) as a basis for our work on the secure evolution of service-oriented architectures.

AWED is available at http://awed.gforge.inria.fr.

5.2. btrCloud (and Entropy)

Participants: Jean-Marc Menaud [correspondent], Rémy Pottier, Clotilde Massot, Guillaume Le Louët, Thierry Bernard, Frédéric Dumont.

Orchestration, virtualization, energy, autonomic system, placement, cloud computing, cluster, data center, scheduler, grid

btrCloud is a virtual machine manager for clusters and provides a complete solution for the management and optimization of virtualized data center. btrCloud (acronym of better cloud) is composed of three parts.

The analysis function enables operatives and people in charge to monitor and analyze how a data-center works, be it on a daily basis or on the long run and predict future trends. This feature includes a performances, an analysis and a trends board.

btrCloud, by the integration of btrScript, provides (semi-)automated, VM lifecycle management, including provisioning, resource pool management, VM tracking, cost accounting, and scheduled deprovisioning. Key features include a thin client interface, template-based provisioning, approval workflows, and policy-based VM placement.

Finally, Several kinds of optimizations are currently available, such as energy and load balancing. The former can help save up to around 20% of the data-center energy consumption, of course depending on the context. The latter enhances provides optimal quality of service for the applications that are hosted in the virtualized data-center.

btrCloud is available at http://www.btrcloud.org.

5.3. ECaesarJ, EJava and EScala

Participants: Jacques Noyé [correspondent], Jurgen Van Ham.

Symmetric AOP, features, software product lines, inheritance, virtual classes, propagating mixin composition, event-based programming, events, declarative events, state machines, CaesarJ, Java, Scala

ECaesarJ is a language developed in the context of the European project AMPLE, as joint work with the *Technische Universität Darmstadt* (TUD). The basic objective was to provide support for directly mapping the high-level features defined by a software product line onto implementation-level features, beyond standard feature-oriented programming. But the language has much wider applications. ECaesarJ can actually be seen as a language which smoothly integrates Object-Oriented Programming, Feature-Oriented Programming, Aspect-Oriented Programming, and Event-based Programming.

It is an extension of Java with *virtual classes* and *propagating mixin composition* (as its ancestor CaesarJ, developed at TUD), but also *declarative events* and *state machines*. Unlike AspectJ, ECaesarJ does not include a class-like concept of aspect. Instead, it deals with pointcuts and pieces of advice as (implicit) events and event handlers, which are standard class members. This makes it possible to use standard inheritance to reuse and refine them. Explicit events can also be used when events must be explicitly triggered as in traditional event-based programming. Finally, in the same way as pointcuts can be composed using logical operators, *declarative events* can be defined as a composition of other events.

This provides a symmetric version of AOP where virtual classes can be used to deal with structural aspects whereas events can be used to deal with behavioral aspects.

In ECaesarJ, a class can also include, as class members, state transitions. Combining this with virtual classes makes it possible to define, at the programming language level, refinable hierarchical state machines. The combination of state machines and events provides, in particular, effective language support for the State design pattern as well as a form of Event-based AOP.

EJava and EScala are more recent developments of the same ideas applied to Java and Scala, respectively. EJava benefits from Java tooling with an eclipse plugin developed with the Spoofax Language Workbench. Unlike EJava and ECaesarJ, EScala makes it possible to dynamically register and unregister event handlers. It also benefits from a more efficient, compiler-based, implementation. As ECaesarJ, EScala is joint work with TUD.

Prototype implementations of these languages are available through http://ecaesarj.gforge.inria.fr/.

5.4. FPath and FScript

Participants: Thomas Ledoux [correspondent], Frederico Alvares.

dynamic reliable reconfiguration, self-adaptive components, Fractal, autonomic computing

FPath and FScript are two domain-specific languages (DSLs) dealing respectively with the navigation and the dynamic reconfiguration of Fractal architectures. *FPath* is a DSL for querying Fractal architectures. It is restricted to the introspection of architectures by browsing elements identified by their properties or location in the architecture. This focused domain allows FPath to offer a very concise and readable syntax and ensures correctness properties by construction (e.g. any query terminates in a finite time). *FScript* is a DSL dedicated to the reconfiguration of Fractal component architectures. It enables reconfiguration scripts to modify a Fractal architecture. Like FPath, FScript guarantees several properties by construction, e.g. termination of scripts by excluding the possibility of infinite loops. Moreover the FScript interpreter supports a transactional model of reconfigurations and the preservation of the ACID properties.

An adaptation of FPath/FScript to FraSCAti, a component framework providing runtime support for the Service Component Architecture (SCA), has been developed by the Inria Adam project-team. In that way, software architects are able to navigate using FPath notation through FraSCAti architectures and to reconfigure them with FScript. We have used this adaptation in our recent work [11][31] for reconfiguring cloud applications in order to reduce the energy footprint in cloud infrastructures.

FScript and its extensions are available under the LGPL license at http://fractal.ow2.org/fscript.

5.5. WildCAT

Participants: Thomas Ledoux [correspondent], Frederico Alvares.

monitoring, context-aware applications, complex event processing

WildCAT is a generic Java framework for context-aware applications. It permits the monitoring of large-scale applications by allowing developers to easily organize and access resources through a hierarchical organization backed with a powerful SQL-like language to inspect sensors data and to trigger actions upon particular conditions. WildCAT proposes two modes to inspect the resources: a pull mode relies on synchronous communication and a push one relies on asynchronous communication. In the pull mode, developers programmatically get and set attributes. In the push mode, developers register listeners on queries expressed over the events generated by the backend.

WildCAT has been developed by the team in the last years. We have used WildCAT in our recent work [11] for allowing cloud applications to listen events notification fired by the cloud infrastructure (e.g. whenever the pricing policy of cloud resources changes) or to detect changes on the application activity (e.g. to detect whenever the number of requests sharply increases/decreases) in order to launch the reconfiguration of cloud applications.

WildCAT is available under GPL v2 at http://wildcat.ow2.org.

ATLANMOD Team

5. Software

5.1. The ATL Model Transformation Language

URL: http://www.eclipse.org/m2m/atl/

With an eye on the normative work of the OMG (MOF, OCL, QVT, etc.), a new conceptual framework has been developed based on a second generation model transformation language called ATL. Although ATL influenced the OMG standard, the approach is more general as discussed in [8]. In 2004 IBM gave an Eclipse innovation award to the ATL project. In 2007 Eclipse recognized ATL as one central solution for model transformation and promoted it to the M2M project (see *Eclipse.org/m2m*). There are more than 200 industrial and academic sites using ATL today, and several Ph.D. thesis in the world are based on this work.

In 2011 we started a new evolution phase for ATL. Our mid-term plan is making of ATL the leading solution for building autonomous reactive transformation systems, i.e. transformation networks that can autonomously manage a set of dataflows among the application models.

Following this line, we first implemented a new refinement mode for ATL, to support in-place transformations. This extension allows the dynamic manipulation of models while keeping them connected to runtime applications. Next, we presented a lazy execution algorithm for ATL. With it, the elements of the target model are generated only when and if they are accessed. This extension allows to build reactive transformation systems that react to requests of model elements, by triggering the necessary computation. Our lazy version of ATL enables also transformations that generate infinite target models, extending the application space of the model-transformation paradigm.

The latest (still ongoing) work in this direction is the development of a full reactive ATL engine, able to activate the minimal computation for responding to updates or request on the involved models. This engine is studied to scale up with large ATL networks. In this line we also introduced an algorithm for simplifying ATL transformation chains.

5.2. MoDisco (Model Discovery)

URL: http://www.eclipse.org/modisco/

MoDisco is an open source Eclipse project that provides a generic and extensible framework dedicated to the elaboration of Model Driven Reverse Engineering (MDRE) solutions. Gathering contributions from both academics and industrials, the goal of the project is to federate common efforts in the model-based transformation of legacy software systems implemented using different technologies (e.g.; Java, COBOL, C). The first principle is to discover models out of legacy artifacts, representing appropriately all the relevant information, to be then used as part of reverse engineering processes for software understanding, evolution or modernization. Targeted scenarios include software (technical or architectural) migration of large legacy systems, but also retro-documentation, refactoring, quality assurance, etc. Within this context, MoDisco has collaborations with the OMG Architecture Driven Modernization (ADM) Task Force, for which the project provides several reference implementations of its standards: Knowledge Discovery Metamodel (KDM), Software Measurement Metamodel (SMM) and Abstract Syntax Tree Metamodel (ASTM).

The MoDisco framework [12] is composed of a set of Eclipse plugins, and relies on the de-facto standard Eclipse Modeling Framework (EMF) for model handling. Thanks to its modular architecture, it allows completely covering the three steps of a standard MDRE approach: 1) Discovery (i.e. extracting a complete model of the source code), 2) Understanding (i.e. browsing and providing views on this model for a given purpose) and 3) Transformation (evolving the model towards a new technology, architecture, etc). More specifically, as part of its *Infrastructure* layer, MoDisco offers the set of generic (i.e.; legacy technology-independent) reusable components really useful to build the core of MDRE solutions: Discovery Manager and Workflow for MDRE task orchestration, Model Browser for advanced navigation in complex models, model extension and customization capabilities for understanding (e.g.; views definition), etc. As part of its *Technologies* layer, it provides an advanced support for the Java, JEE and XML technologies, including complete metamodels, corresponding model discoverers, transformations, code generators, customizations, query libraries, etc.

MoDisco (or some of its components) is being used by different partners including other academics, industrials (e.g.; Sodifrance on several of their real modernization projects for their customers) or Eclipse projects (e.g.; Eclipse-MDT Papyrus as developed by CEA). Moreover, the Eclipse-EMFT EMF Facet project has been initiated as a MoDisco spin-off, in order to externalize some features which are not actually specific to reverse engineering problems and thus may be reused in many different contexts (cf. corresponding EMF Facet section).

The initiative continues to be developed within the context of the European FP7-ICT project named ARTIST [2], and also to a lower extent within the context of the French FUI 13 project named TEAP.

5.3. Community-driven language development

URL: http://code.google.com/a/eclipselabs.org/p/collaboro/

Software development processes are collaborative in nature. Neglecting the key role of end-users leads to software that does not satisfy their needs. This collaboration becomes specially important when creating Domain-Specific Languages (DSLs), which are (modeling) languages specifically designed to carry out the tasks of a particular domain. While end-users are actually the experts of the domain for which a DSL is developed, their participation in the DSL specification process is still rather limited nowadays.

Thus, Collaboro is an approach to make language development processes more participative, meaning that both developers and users of the language can collaborate together to create and evolve it. The tool has been developed as an Eclipse plugin, whose features currently implemented are:

- Version view to navigate through the Proposals of a version. For each Proposal, the solutions and comments are shown.
- Collaboration view to show the data related to a Collaboration selected in the version view. This view also shows the changes to apply if the selected element is a Solution.
- The user can login to the Collaboro system and create proposals, solutions and comments by right-clicking in the version view. The user can also vote for/against the collaborations.
- Decision engine based on a total agreement (i.e., all the community users must vote for the collaboration). The decision engine can be launch by using the bar menu.
- Notation engine and Notation view to render SVG snapshots of the DSL concrete syntax.

5.4. Virtual EMF (Model Virtualization)

URL: http://code.google.com/a/eclipselabs.org/p/virtual-emf/

[2]http://www.artist-project.eu/

Virtual EMF is an Eclipse plugin built on top of EMF that enables the creation and manipulation of *virtual models*, i.e., models whose elements do not contain concrete data, but are rather proxies to elements contained in other models. The idea is related to that of model composition, as it aims capturing the (often overlapping) concepts as one single global model. This is a frequently faced problem as, in complex scenarios, modelers often have to deal with a large number of heterogeneous and interrelated models. Most times, the view a specific kind of user requires does not correspond to any of these models but is a combination of cross-domain information scattered among several ones.

Current composition techniques rely on the materialization of the composed model, an approach that poses some important limitations in terms of (i)*efficiency*, as they do not scale (the data duplication mechanism they use implies in extra memory usage and time-consuming generation of the composed model), (ii)*synchronization*, as updates in the composed model are not propagated to the original ones (or vice-versa) thus losing consistency, or even (iii)*interoperability*, as in some cases the composed model requires a specific API/tool to be handled.

Virtual EMF allows overcoming the limitations above. A virtual model provides to tools/users the *illusion* of working with a regular model whereas, in fact, all model access and manipulation requests are transparently redirected to its set of *virtualized* models. It serves as a centralized and transparent access point to a set of interconnected models, allowing users to easily compose, weave and link them. It provides the following beneficial properties:

- *Interoperability:* it behaves as a normal model. Therefore, compatibility with existing EMF-based solutions/tools (e.g. models transformations, model editors, ...) is guaranteed;
- *Synchronization:* changes are automatically and transparently propagated between virtual and original models;
- *Scalability:* support for very big models;
  - low memory usage: no data duplication, direct access to original model elements;
  - faster generation time: no need for (time-consuming) information cloning operations (e.g. executing a model transformation);
- *Genericity:* support for several types of inter-model relationships (e.g. merge, association, filter) and extension capabilities for their semantics.

Virtual EMF is available as an open source project on Eclipse Labs. It has been contributed by the AtlanMod team to the CESAR project. The initiative continues to be developed within the context of the French FUI 13 project named TEAP [TODO Put ref to TEAP http://www.atlanpole.fr/Atlanpole-Digital-Innovation/liste-des-news/TEAP-Projet-Collaboratif-d-Innovation].

5.5. EMFtoCSP

URL: http://code.google.com/a/eclipselabs.org/p/emftocsp/

EMFtoCSP is a tool for the verification of precisely defined conceptual models and metamodels. For these models, the definition of the general model structure (using UML or EMF) is supplemented by OCL constraints. The Eclipse Modeling Development Tools (MDT [3]) provides mature tool support for such OCL-annotated models with respect to model definition, transformation, and validation.

However, an additional important task that is not supported by Eclipse MDT is the assurance of model quality. A systematical assessment of the correctness of such models is a key issue to ensure the quality of the final application. EMFtoCSP fills this gap by provided support for automated model verification in Eclipse.

[3] http://www.eclipse.org/modeling/mdt/?project=ocl

Essentially, the EMFtoCSP is a sophisticated bounded model finder that yields instances of the model that conform not only to the structural definition of the model (e.g., the multiplicity constraints), but also to the OCL constraints. Based on this core, several correctness properties can be verified:

1. Satisfiability – is the model able to express our domain? For this check, the minimal number of instances and links can be specified to ensure non-trivial instances.
2. Unsatisfiability – is the model unable to express undesirable states? To verify this, we add further constraints to the model that state undesired conditions. Then we can check if is it impossible to instantiate the amended model.
3. Constraint subsumption – is one constraint already implied by others (and could therefore be removed)?
4. Constraint redundancy – do different constraints express the same fact (and could therefore be removed)?

To solve these search problems, EMFtoCSP translates the EMF/OCL (resp. UML/OCL) model into a constraint satisfaction problem and employs the Eclipse CLP solver [4] to solve it. This way, constraint propagation is exploited to tackle the (generally NP-hard) search.

The tool is a continuation of the UMLtoCSP approach [48] developed previously by Jordi Cabot, Robert Clarisó and Daniel Riera. It provides a generic plugin framework for Eclipse to solve OCL-annotated models using constraint logic programming. Apart from already supported Ecore and UML metamodels, further metamodels can be added easily in the future. Similarly, other constraint solving back-ends can be integrated. It is provided under the Eclipse Public License.

5.6. EMF Facet

URL: http://www.eclipse.org/modeling/emft/facet/

EMF Facet is an open source Eclipse project, under the Eclipse Public License (EPL), that provides a generic and extensible framework dedicated to the dynamic and non-intrusive extension of models. It can be used to extend already existing metamodels with additional concepts and properties, the corresponding models being then transparently augmented, reduced or modified accordingly at runtime. Such a metamodel extension is called a facet, and can be specified on top of any metamodel in EMF Ecore. The underlying mechanism is based on the runtime execution of queries on the models corresponding to the *faceted* metamodels. Facets are notably particularly relevant for obtaining different views on existing models without having to actually alter them with any extra data.

The EMF Facet framework is composed of several Eclipse plugins, and relies on the de-facto standard Eclipse Modeling Framework (EMF) for model handling. The facet definitions are stored as facet models, allowing them to be exchanged and reused in various contexts. The queries can be implemented using any suitable query language (e.g.; ATL, OCL, Java, XPath), as far as the corresponding adapters exist and are correctly registered within the framework. The proposed tooling includes dedicated editors for creating, editing and saving both facet and query definitions, the implemented support for Java, OCL and ATL queries, a Table Editor for visualizing query results. An advanced support for the model display customization (e.g.; icons, colors, fonts) is also provided as part of the framework.

EMF Facet is currently intensively used in MoDisco for extracting and displaying different specific views from large models of legacy systems. Its extension and customization capabilities are actually integrated into several MoDisco components, such as notably the MoDisco Model Browser. However, different other integration possibilities will be also explored in the future.

The initiative continues to be developed within the context of the European FP7-ICT project ARTIST.

[4] http://eclipseclp.org/

5.7. Industrialization strategy for research prototypes

Research labs, as a source of innovation, are potential key actors of the Software Engineering market. However, an important collaborative effort with the other players in the software industry is still needed in order to actually transfer the corresponding techniques or technologies from the research lab to a company. Based on the AtlanMod concrete experience with the previously mentioned open source tools/projects, we have extracted a pragmatic approach [3] for transforming the results of scientific experimentation into practical industrial solutions.

While dealing with innovation, this approach is also innovation-driven itself, as the action is actually conducted by the research lab via a technology transfer. Three different partners are directly involved in this process, using open source as the medium for maintaining a constant interaction between all of them:

- **Use Case Provider.** Usually a company big enough to have to face real complex industrial scenarios which need to be solved (at least partially) by applying new innovative principles and techniques;
- **Research Lab.** Usually a group from a research institute (public or private) or university evaluating the scientific relevance of the problems, identifying the research challenges and prototyping possible solutions;
- **Technology Provider.** Usually a small or medium company, with a particular technical expertise on the given domain or Software Engineering field, building and delivering the industrial version of the designed solutions;

From our past and current experience, three main characteristics of this industrialization *business model* can be highlighted:

- **Win-win situation.** Each partner can actually focus on its core activity while also directly benefiting from the results obtained by the others (notably the research lab can continue to do research);
- **Application-driven context.** The end-user need is at the origin of the process, which finally makes the developed solution actually relevant;
- **Iterative process.** The fact of having three distinct partners requires different regular and consecutive exchanges between all of them.

CIDRE Project-Team

5. Software

5.1. Intrusion Detection

Members of Supélec have developed several intrusion detectors.

Blare implements our approach of illegal information flow detection at the OS level. This implementation is a modification of a standard Linux kernel and it monitors information flows between typical OS containers as files, sockets or IPC. System active entities are processes viewed as black-boxes as we only observe their inputs and outputs. Detection at the OS level is in some cases too coarse-grained to avoid the generation of false positives and to detect attacks targeting the application logic. Even if it remains convenient to define the security policy at the OS-level, sound illegal information flow detection implies an additional detection at the language level. This has led us to implement a detector for Java applications, **JBlare**, to complement the detection at the OS level. JBlare extends the OS-level one by refining the observation of information flows at the language level.

GNG is an intrusion detection system that correlates different sources (such as different logs) in order to identify attacks against the system. The attack scenarios are defined using the Attack Description Langage (**ADeLe**) proposed by our team, and are internally translated to attack recognition automatons. GNG intends to define time efficient algorithms based on these automatons to recognize complex attack scenarios.

SIDAN (Software Instrumentation for Detecting Attacks on Non-control-data) is a tool that aims to instrument automatically C-language software with assertions whose role is to detect attacks against the software. This tool is implemented as a plugin of the FRAMA-C framework that provides an implementation of static analysis techniques.

5.2. Privacy

GEPETO (GEoPrivacy-Enhancing TOolkit) is an open source software for managing geolocated data (currently in development in cooperation with LAAS). GEPETO can be used to visualize, sanitize, perform inference attacks and measure the utility of a particular geolocated dataset. For each of these actions, a set of different techniques and algorithms can be applied. The global objective of GEPETO is to enable a user to design, tune, experiment and evaluate various sanitization algorithms and inference attacks as well as visualizing the following results and evaluating the resulting trade-off between privacy and utility. An engineer (Izabela Moise) is currently working on the development of a distributed version of GEPETO based on the MapReduce paradigm and the Hadoop framework, in order to make it able to deal with datasets composed of millions of mobility traces.

FOCUS Project-Team

5. Software

5.1. Jolie

Members of Focus have developed Jolie [8] (Java Orchestration Language Interpreter Engine, see http://www.jolie-lang.org/). Jolie is a service-oriented programming language. Jolie can be used to program services that interact over the Internet using different communication protocols. Differently from other Web Services programming languages such as WS-BPEL, Jolie is based on a user-friendly C/Java-like syntax (more readable than the verbose XML syntax of WS-BPEL) and, moreover, the language is equipped with a formal operational semantics. This language is used for the *proof of concepts* developed around Focus activities. For instance, contract theories can be exploited for checking the conformance of a Jolie program with respect to a given contract. A spin-off, called "Italiana Software", has been launched around Jolie, its general aim is to transfer the expertise in formal methods for Web Services matured in the last few years onto Service Oriented Business Applications. The spin-off is a software producer and consulting company that offers service-oriented solutions (for instance, a "'single sign-on" application) based on the Jolie language.

In 2012 the development of Jolie has continued. The main activities have been:

- We have enhanced the correlation mechanism in Jolie to handle multiparty sessions with concurrent interactions with multiple participants.
- We have developed a compiler that projects choreography-based programs in Chor (http://www.chor-lang.org) to Jolie.
- We have developed a new website for Jolie (using Jolie itself), significantly updating its documentation.
- We have improved Jolie's compatibility with the Java RMI technology.
- We have developed a first experimental implementation of a monitoring layer for Jolie services.

As last year, so in 2012 Jolie has been used for teaching, in a master course at the IT University of Copenhagen (ITU, Denmark) and in a master course at the Technical University of Denmark (DTU, Denmark).

5.2. Others

Below we list some sofwtare that has been developed, or is under development, in Focus.

- *IntML* is a functional programming language guaranteeing sublinear space bounds for all programs [53]. See the Activity Reports of previous years (in particular 2010) for more details. During 2012 no substantial modifications have been made.
- *Lideal* (http://lideal.cs.unibo.it/) is an experimental tool implementing type inference for dependently linear type systems. The tool reduces the problem of evaluating the complexity of PCF (i.e. functional programs with primitive integers and recursive definitions) to checking a set of first-order inequalities for validity. The latter can then be handled through SMT solvers or put in a form suitable for managing them with tools such as CoQ.
- We have implemented a technique for the deadlock analysis of a concurrent object oriented language (ABS, designed within the European project HATS). The technique consists of
  - an inference system for contracts to be associated to methods. Contracts are terms that retain information about resource dependencies;
  - a fixpoint algorithm for solving contract definitions, which are recursive and may introduce new resource names.

The release of the software is planned for early 2013.

- *Croll-pi Interpreter* (http://proton.inrialpes.fr/~mlienhar/croll-pi/implem/). We have developed an interpreter for croll-pi using Maude. Croll-pi is a concurrent reversible language featuring a rollback operator to undo a past action (together with all the actions depending on it), and a compensation mechanism to avoid cycling by redoing the same action again and again.

  We used the interpreter to test the expressive power of croll-pi on various problems, including the 8-queen problem, error handling in an automotive scenario from the EU project Sensoria, and constructs for distributed error handling such as stabilizers.

For other software, such as `PiDuce`, see the activity reports for Focus of previous years.

INDES Project-Team

5. Software

5.1. Introduction

Most INDES software packages, even the older stable ones that are not described in the following sections are freely available on the Web. In particular, some are available directly from the Inria Web site:

`http://www.inria.fr/valorisation/logiciels/langages.fr.html`

Most other software packages can be downloaded from the INDES Web site:

`http://www-sop.inria.fr/teams/indes`

5.2. Functional programming

Participants: Frédéric Boussinot [Inria], Cyprien Nicolas [Inria], Bernard Serpette [Inria], Manuel Serrano [correspondant].

5.2.1. The Bigloo compiler

The programming environment for the Bigloo compiler [5] is available on the Inria Web site at the following URL: http://www-sop.inria.fr/teams/indes/fp/Bigloo. The distribution contains an optimizing compiler that delivers native code, JVM bytecode, and .NET CLR bytecode. It contains a debugger, a profiler, and various Bigloo development tools. The distribution also contains several user libraries that enable the implementation of realistic applications.

BIGLOO was initially designed for implementing compact stand-alone applications under Unix. Nowadays, it runs harmoniously under Linux and MacOSX. The effort initiated in 2002 for porting it to Microsoft Windows is pursued by external contributors. In addition to the native back-ends, the BIGLOO JVM back-end has enabled a new set of applications: Web services, Web browser plug-ins, cross platform development, etc. The new BIGLOO .NET CLR back-end that is fully operational since release 2.6e enables a smooth integration of Bigloo programs under the Microsoft .NET environment.

5.2.2. The FunLoft language

FunLoft (described in http://www-sop.inria.fr/teams/indes/rp/FunLoft) is a programming language in which the focus is put on safety and multicore.

FunLoft is built on the model of FairThreads which makes concurrent programming simpler than usual preemptive-based techniques by providing a framework with a clear and sound semantics. FunLoft is designed with the following objectives:

- provide a safe language, in which, for example, data-races are impossible.
- control the use of resources (CPU and memory), for example, memory leaks cannot occur in FunLoft programs, which always react in finite time.
- have an efficient implementation which can deal with large numbers of concurrent components.
- benefit from the real parallelism offered by multicore machines.

A first experimental version of the compiler is available on the Reactive Programming site http://www-sop.inria.fr/teams/indes/rp. Several benchmarks are given, including cellular automata and simulation of colliding particles.

5.3. Web programming

Participants: Gérard Berry [Inria], Cyprien Nicolas [Inria], Manuel Serrano [correspondant].

5.3.1. The HOP web programming environment

HOP is a higher-order language designed for programming interactive web applications such as web agendas, web galleries, music players, etc. It exposes a programming model based on two computation levels. The first one is in charge of executing the logic of an application while the second one is in charge of executing the graphical user interface. HOP separates the logic and the graphical user interface but it packages them together and it supports strong collaboration between the two engines. The two execution flows communicate through function calls and event loops. Both ends can initiate communications.

The HOP programming environment consists in a web *broker* that intuitively combines in a single architecture a web server and a web proxy. The broker embeds a HOP interpreter for executing server-side code and a HOP client-side compiler for generating the code that will get executed by the client.

An important effort is devoted to providing HOP with a realistic and efficient implementation. The HOP implementation is *validated* against web applications that are used on a daily-basis. In particular, we have developed HOP applications for authoring and projecting slides, editing calendars, reading RSS streams, or managing blogs.

HOP has won the software *open source contest* organized by the ACM Multimedia Conference 2007 http://mmc36.informatik.uni-augsburg.de/acmmm2007/. It is released under the GPL license. It is available at http://hop.inria.fr.

5.4. Language-based security

Participants: Zhengqin Luo [Inria], Tamara Rezk [correspondant].

5.4.1. CFlow

The prototype compiler “CFlow” takes as input code annotated with information flow security labels for integrity and confidentiality and compiles to F# code that implements cryptography and protocols that satisfy the given security specification.

Cflow has been coded in F#, developed mainly on Linux using mono (as a substitute to .NET), and partially tested under Windows (relying on .NET and Cygwin). The code is distributed under the terms of the CeCILL-B license http://www.msr-inria.inria.fr/projects/sec/cflow/index.html.

5.4.2. FHE type-checker

We have developed a type checker for programs that feature modern cryptographic primitives such as fully homomorphic encryption. The type checker is thought as an extension of the “CFlow” compiler developed last year on the same project. It is implemented in F#. The code is distributed under the terms of the CeCILL-B license http://www.msr-inria.inria.fr/projects/sec/cflow/index.html.

5.4.3. Mashic compiler

The Mashic compiler is applied to mashups with untrusted scripts. The compiler generates mashups with sandboxed scripts, secured by the same origin policy of the browsers. The compiler is written in Bigloo and can be found at http://www-sop.inria.fr/indes/mashic/.

5.5. Old software

5.5.1. Camloo

Camloo is a caml-light to bigloo compiler, which was developed a few years ago to target bigloo 1.6c. New major releases 0.4.x of camloo have been done to support bigloo 3.4 and bigloo 3.5. Camloo make it possible for the user to develop seamlessly a multi-language project, where some files are written in caml-light, in C, and in bigloo. Unlike the previous versions of camloo, 0.4.x versions do not need a modified bigloo compiler to obtain good performance. Currently, the only supported backend for camloo is bigloo/C. We are currently rewriting the runtime of camloo in bigloo to get more portability and to be able to use HOP and camloo together.

5.5.2. Skribe

SKRIBE is a functional programming language designed for authoring documents, such as Web pages or technical reports. It is built on top of the SCHEME programming language. Its concrete syntax is simple and looks familiar to anyone used to markup languages. Authoring a document with SKRIBE is as simple as with HTML or LaTeX. It is even possible to use it without noticing that it is a programming language because of the conciseness of its original syntax: the ratio *tag/text* is smaller than with the other markup systems we have tested.

Executing a SKRIBE program with a SKRIBE evaluator produces a target document. It can be HTML files for Web browsers, a LaTeX file for high-quality printed documents, or a set of *info* pages for on-line documentation.

5.5.3. Scheme2JS

Scm2JS is a Scheme to JavaScript compiler distributed under the GPL license. Even though much effort has been spent on being as close as possible to R5RS, we concentrated mainly on efficiency and interoperability. Usually Scm2JS produces JavaScript code that is comparable (in speed) to hand-written code. In order to achieve this performance, Scm2JS is not completely R5RS compliant. In particular it lacks exact numbers.

Interoperability with existing JavaScript code is ensured by a JavaScript-like dot-notation to access JavaScript objects and by a flexible symbol-resolution implementation.

Scm2JS is used on a daily basis within HOP, where it generates the code which is sent to the clients (web-browsers). Scm2JS can be found at `http://www-sop.inria.fr/indes/scheme2js`.

LOGNET Team

4. Software

4.1. myMed

Our flagship software is called myMed. myMed is a highly innovative project in which three main orthogonal components are brought together:

- a software development kit, SDKmyMed, with which we can build social networks in "rush time";
- a novel distributed hosting cloud, CLOUDmyMed, with which the social applications (developed by us and by third parties) can be hosted and run;
- a pull of 5-10 social network applications, aka "sociapps" developed in our team to test the SDKmyMed.

The sociapp can be enjoyed in almost all platforms, from web browsers, to mobile web, until IOS and Android devices.

4.2. myMed backbone

Participants: Luigi Liquori [contact], The myMed Engineer Team.

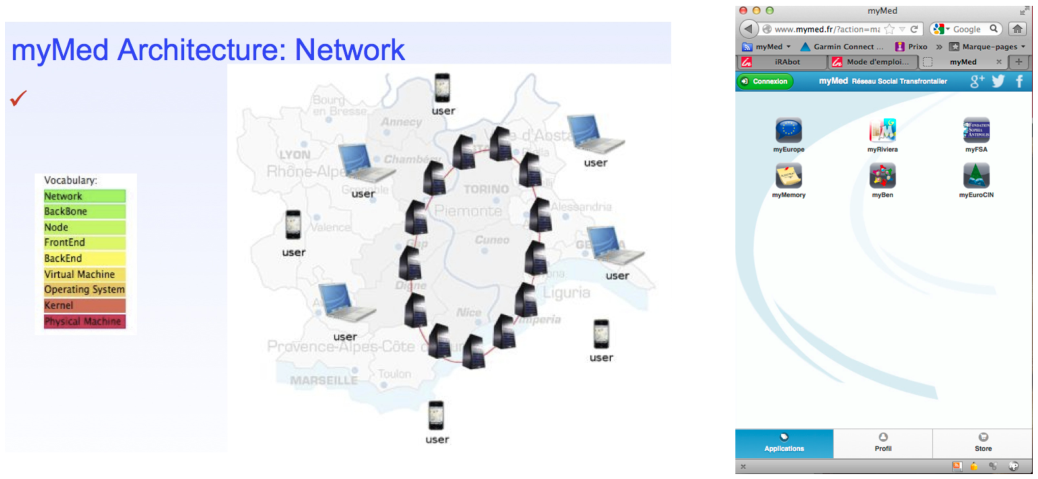


Figure 4. The myMed backbone and the myMed LaunchPad

We have implemented a "backbone" for the myMed social network using a nosql database called Cassandra http://cassandra.apache.org, the latter used also by social networks like Facebook and Twitter. The backbone relies on 50 PC quad code HP400, equipped with 2Tb of hard drive each.

4.3. myMed frontend

Participants: Luigi Liquori [contact], The myMed Engineer Team.

We have implemented a front-end with which all the social application can be used and downloaded via a “store” mechanism similar to the ones of Apple and Google stores. Social applications can be chosen, voted for via a reputation system, and uninstalled (including all personal data) if the user wants. We have also implemented a “template” allowing to build “proofs-of-concept” of social networks in a very short time.

4.4. Synapse simulator in Oversim

Participant: Vincenzo Ciancaglini [contact].

Synapse-Oversim is an implementation of the Synapse overlay interconnection protocol in the Oversim overlay simulator. The software presents two main contributions: first of all, a fork of the original Oversim simulator has been implemented in order to support running multiple protocol modules in a single instance of Oversim, a necessary feature in order to simulate a set of heterogeneous interconnected networks. Secondly, the whole Synapse protocol has been implemented on top of Oversim, in order to allow for the efficient inter-routing of messages between heterogeneous overlays. The Synapse code has been developed in C++, by running in Oversim, its correctness and its performances can be evaluated, while then the code can be easily ported to a real-world application.

4.5. Synapse model Erlang validator

Participant: Vincenzo Ciancaglini [contact].

During the work on the Synapse protocol, we devised a mathematical model which would allow us to estimate performance indexes of an interconnected system without having to deploy a full-scale experiment. In order to be validated, however, the model results needed to be verified against some simulation results, run under simplified conditions, but with the highest possible number of nodes. To achieve this, a dedicated simulator has been developed using Erlang, a programming language dedicated to parallel and distributed applications, which allow for the simulation of extreme systems, with a number of nodes beyond one million, in the fastest way achievable, by fully exploiting the multicore architecture of modern machines. The simulator instantiates a lightweight thread for each node, and the communication are rendered by message passing between the different node threads, thus keeping the simulation conditions as close as possible to a real world behavior.

4.6. CCN-TV Omnet++ simulator

Participant: Vincenzo Ciancaglini [contact].

CCN-TV-SIM is a software, based on the network simulation framework Omnet++, which simulates a real time video broadcast system over content-centric networks. The system is able to manage multiple streams of video at different rates, using real video traces, simulate different caching policies, different channels being transmitted concurrently, background network traffic, and different channel switch rates. Furthermore it can exploits network topologies taken from real networks, like the Deutsche Telecom network, or the Geant.

4.7. Java implementation of the OGP protocol and the experiment controller

Participant: Hoang Giang Ngo [contact].

OGP-Experiment contains Java implementation of the OGP protocol (OGP stands for overlay gateway protocol) which is used for inter-routing between heterogeneous overlay networks, and a Java implementation of the experiment controller, which is responsible for scheduling, managing and monitoring the statistics of the experiments. The software supports experiments in churn and no-churn environments. Performance metrics of the OGP protocol, such as the latency, the successful rate of data lookup and the traffic generated by a peer are reported. The experiments are performed on the Grid 5000 platform. Heterogeneous overlays which are connected by OGP can be easily plugged into the software.

4.8. Java implementation of the Synapse protocol and the experiment controller

Participant: Hoang Giang Ngo [contact].

Synapse-Experiment contains Java implementation of the Synapse overlay interconnection protocol and Java implementation of the experiment controller which is responsible for scheduling, managing and monitoring the statistics of the experiments. The software supports experiments in churn and no-churn environments. Performance metrics of the Synapse protocol, such as the latency, the successful rate of data looking up and the traffic generated by a peer are reported. The experiments are performed on the Grid 5000 platform.

4.9. Reputation Computation Engine for Social Web Platforms

Participants: Thao Nguyen [contact], Laurent Vanni.

Among the three components of a Trust and Reputation System, information gathering is most dependent on the application system, followed by the decision support component and then by the building of a robust Reputation Computation Engine and an experimental GUI, showing how bad users are segregated by the engine. To simulate the working of the reputation engine, we set up a population of Nu users, providing the same service, and undertaking Nt transactions. In each transaction, a random consumer is assigned to request the service. Other users will then be candidate providers for this request. When a user plays the role of a consumer, his behavior is modeled in the raterType attribute. Three types of raters include HONEST, DISHONEST and COLLUSIVE. HONEST raters share their personal experience honestly, i.e. Rr = Ep. DISHONEST raters provide ratings 0:5 different from their true estimation, i.e. Rr = Ep +- 0:5. COLLUSIVE raters give the highest ratings (Rr = 1) to users in their collusion and the lowest ratings (Rr = 0) to the rest. Similarly, when a user acts as a provider, he can be one of the following types of providers: GOOD, NORMAL, BAD, or GOODTURNBAD. This type is denoted in providerType attribute. The QoS of the service provided by a BAD, NORMAL, or GOOD provider has a value in the interval (0; 0:4], (0:4; 0:7], or (0:7; 1] respectively. A GOODTURNBAD provider will change the QoS of his service when 50% of Nt transactions have been done in the simulation. To get a transaction done, a consumer obtains a list of providers, computes reputation scores for them, chooses a provider to perform the transaction, updates his private information, and publishes his rating for the provider. The quality of service that the consumer will experience depends on the providerType of the chosen provider. The difference between the consumer's rating for the provider and his observation depends on the consumer's raterType.

To run a simulation, the user must specify 10 parameters as described above: Simulation(Nu, Nt, %G, %N, %B, %GTB, %H, %D, %C, %dataLost). The simulator has been published in [22].

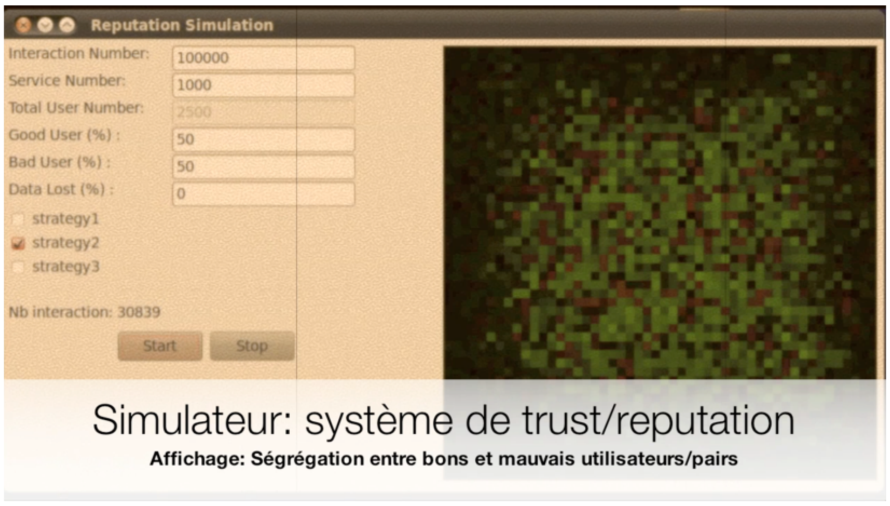
4.10. Ariwheels

Participants: Luigi Liquori [contact for the *Ariwheels* simulator], Claudio Casetti [Politecnico di Torino, Italy], Diego Borsetti [Politecnico di Torino, Italy], Carla-Fabiana Chiasserini [Politecnico di Torino, Italy], Diego Malandrino [Politecnico di Torino, Italy, contact for the *Ariwheels* client].

Ariwheels is an info-mobility solution for urban environments, with access points deployed at both bus stops (forming thus a wired backbone) and inside the buses themselves. Such a network is meant to provide connectivity and services to the users of the public transport system, allowing them to exchange services, resources and information through their mobile devices. *Ariwheels* is both:

- a protocol, based on *Arigatoni* and the publish/subscribe paradigm;
- a set of applications, implementing the protocol on the different types of nodes;
- a simulator, written in OMNET++ and recently ported to the ns2 simulator, see Fig 6 .

See the web page http://www-sop.inria.fr/members/Luigi.Liquori/ARIGATONI/Ariwheels.htm and http://arigtt.altervista.org.


Reputation Simulation
strategy1
strategy2
strategy3
Start
Stop
Simulateur: système de trust/reputation
Affichage: Ségrégation entre bons et mauvais utilisateurs/pairs

Interaction Number: 100000
Service Number: 1000
Total User Number: 2500
Good User (%) : 50
Bad User (%) : 50
Data Lost (%) : 0
Nb interaction: 30839

Figure 5. A prototype graphical GUI for the Reputation Computation Engine

4.11. Arigatoni simulator

Participants: Luigi Liquori [contact], Raphael Chand [Université de Geneva, Switzerland].

We have implemented in C++ ($\sim$2.5K lines of code) the Resource Discovery Algorithm and the Virtual Intermittent Protocol of the Arigatoni Overlay Network. The simulator was used to measure the load when we issued n service requests at Global Computers chosen uniformly at random. Each request contained a certain number of instances of one service, also chosen uniformly at random. Each service request was then handled by the Resource Discovery mechanism of Arigatoni networks.

4.12. Synapse client

Participants: Laurent Vanni [contact], Luigi Liquori, Cédric Tedeschi, Vincenzo Ciancaglini.

In order to test our Synapse protocol [17] on real platforms, we have initially developed JSynapse, a Java software prototype, which uses the Java RMI standard for communication between nodes, and whose purpose is to capture the very essence of our Synapse protocol. It is a flexible and ready-to-be-plugged library which can interconnect any type of overlay networks. In particular, JSynapse fully implements a Chord-based inter-overlay network. It was designed to be a lightweight and easy-to-extend software. We also provided some practical classes which help in automating the generation of the inter-overlay network and the testing of specific scenarios. We have experimented with JSynapse on the Grid'5000 platform connecting more than 20 clusters on 9 different sites. Again, Chord was used as the intra-overlay protocol. See, http://www-sop.inria.fr/teams/lognet/synapse-net2012/.

4.13. Open Synapse client

Participant: Bojan Marinkovic [contact].

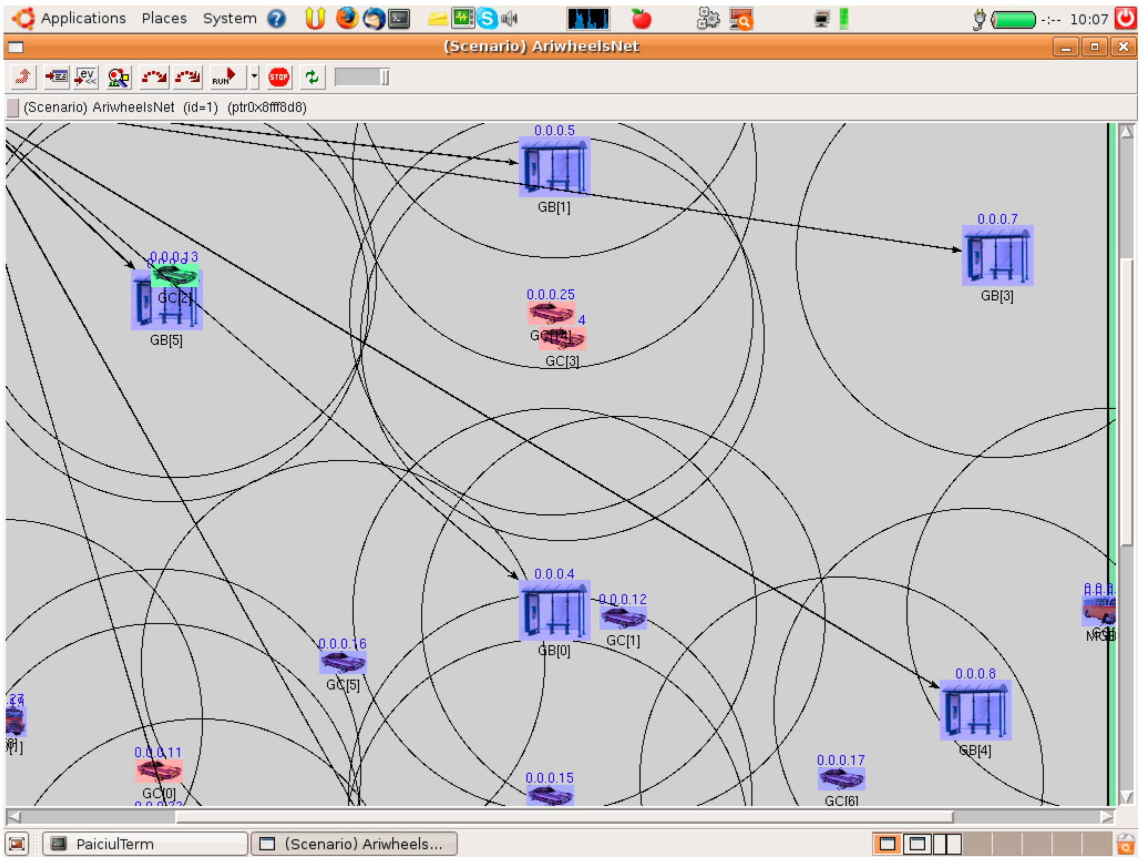
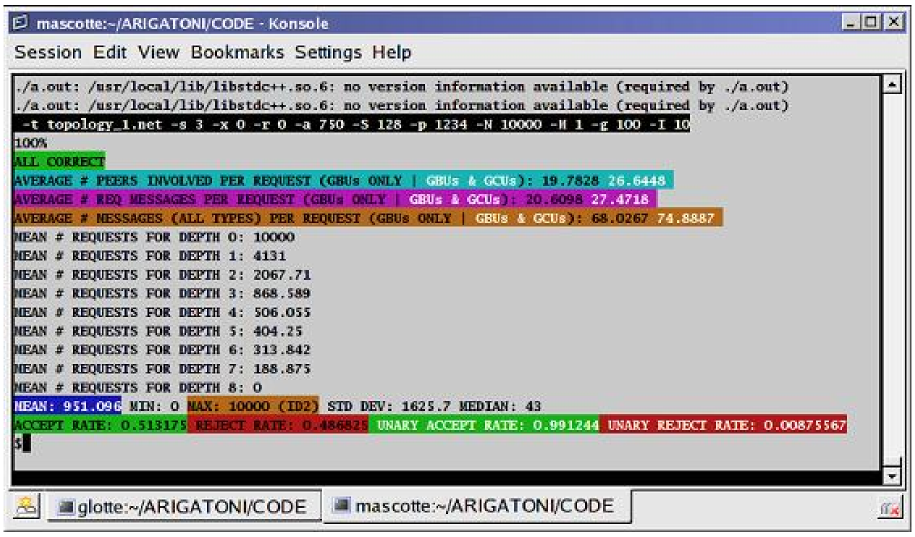


Figure 6. The Ariwheels simulator in Omnet


mascotte:~/ARIGATONI/CODE - Konsole
Session Edit View Bookmarks Settings Help
glotte:~/ARIGATONI/CODE
mascotte:~/ARIGATONI/CODE

```
./a.out: /usr/local/lib/libstdc++.so.6: no version information available (required by ./a.out)
./a.out: /usr/local/lib/libstdc++.so.6: no version information available (required by ./a.out)
 -t topology_1.net -s 3 -x 0 -r 0 -a 750 -S 128 -p 1234 -N 10000 -H 1 -g 100 -I 10
100%
ALL CORRECT
AVERAGE # PEERS INVOLVED PER REQUEST (GBUs ONLY | GBUs & GCUs): 19.7828 26.6448
AVERAGE # REQ MESSAGES PER REQUEST (GBUs ONLY | GBUs & GCUs): 20.6098 27.4718
AVERAGE # MESSAGES (ALL TYPES) PER REQUEST (GBUs ONLY | GBUs & GCUs): 68.0267 74.8887
MEAN # REQUESTS FOR DEPTH 0: 10000
MEAN # REQUESTS FOR DEPTH 1: 4131
MEAN # REQUESTS FOR DEPTH 2: 2067.71
MEAN # REQUESTS FOR DEPTH 3: 868.589
MEAN # REQUESTS FOR DEPTH 4: 506.055
MEAN # REQUESTS FOR DEPTH 5: 404.25
MEAN # REQUESTS FOR DEPTH 6: 313.842
MEAN # REQUESTS FOR DEPTH 7: 188.875
MEAN # REQUESTS FOR DEPTH 8: 0
MEAN: 951.096 MIN: 0 MAX: 10000 (ID2) STD DEV: 1625.7 MEDIAN: 43
ACCEPT RATE: 0.513175 REJECT RATE: 0.486825 UNARY ACCEPT RATE: 0.991244 UNARY REJECT RATE: 0.00875567
$
```

Figure 7. The Arigatoni simulator

Opensynapse is an open source implementation of [17]. It is available for free under the GNU GPL. This implementation is based on Open Chord (v. 1.0.5) - an open source implementation of the Chord distributed hash table implementation by Distributed and Mobile Systems Group Lehrstuhl fuer Praktische Informatik Universitaet Bamberg, see http://www-sop.inria.fr/teams/lognet/synapse-net2012/.

Opensynapse is implemented on top of an arbitrary number of overlay networks. Inter-networking can be built on top of Synapse in a very efficient way. Synapse is based on co-located nodes playing a role that is reminiscent of neural synapses. The current implementation of Opensynapse in this precise case interconnects many Chord overlay networks. The new client currently can interconnect an arbitrary number of Chord networks. This implementation follows the notation presented in [16], and so, each new Chord network is called a *Floor*.

4.14. Husky interpreter

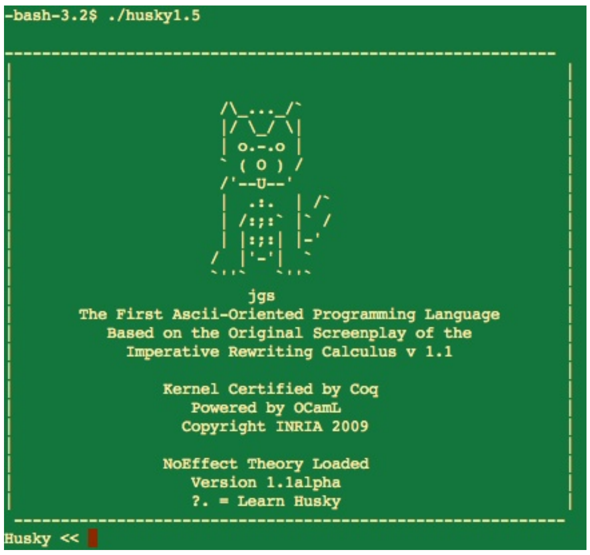
Participants: Marthe Bonamy [contact], Luigi Liquori.

Husky is a variable-less language based on lambda calculus and term rewriting systems. Husky is based on the version 1.1 of *Snake* [13]. It was completely rewritten in CAML by Marthe Bonamy, ENSL (new parser, new syntactic constructions, like, *e.g.*, guards, anti-patterns, anti-expressions, exceptions and parametrized pattern matching). In *Husky*, all the keywords of the language are ASCII-symbols. It could be useful for teaching basic algorithms and pattern-matching to children.

4.15. myTransport Gui

Participants: Laurent Vanni [contact], Vincenzo Ciancaglini, Liquori Liquori.

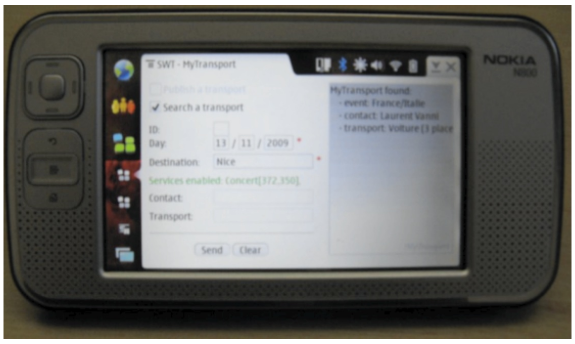
myTransport is a GUI built on top of the Synapse protocol and network. Its purpose is to be a proof of concept of the future service of info-mobility to be available in the myMed social Network, see Figure 9 . The GUI is written in Java and it is fully functional in the Nokia N800 Internet tablet devices. myTransport has been ported to the myMed social network.



```
-bash-3.2$ ./husky1.5

-----------------------------------------------------------
|                                                          |
|                                                          |
|                      /\_..._/`                           |
|                      |/ \_/ \|                           |
|                      | o.-.o |                           |
|                      ` ( O ) /                           |
|                      /'--U--'                            |
|                      |  .:.  | /`                        |
|                      | /:;:` |` /                        |
|                      | |:;:| |-'                         |
|                     /  |'-'| `                           |
|                     `''`    `''`                         |
|                          jgs                             |
|        The First Ascii-Oriented Programming Language     |
|           Based on the Original Screenplay of the        |
|             Imperative Rewriting Calculus v 1.1          |
|                                                          |
|                 Kernel Certified by Coq                  |
|                    Powered by OCamL                      |
|                   Copyright INRIA 2009                   |
|                                                          |
|                 NoEffect Theory Loaded                   |
|                    Version 1.1alpha                      |
|                    ?. = Learn Husky                      |
 -----------------------------------------------------------
Husky <<
```

Figure 8. Launching the Husky interpreter

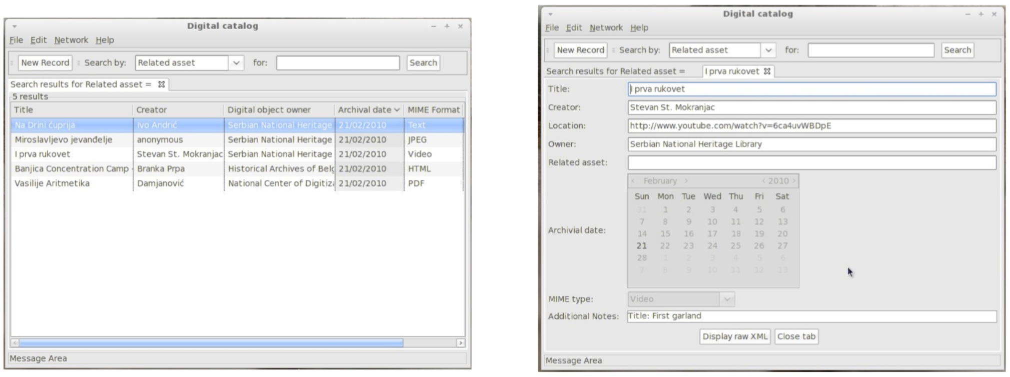


[✓ Search a transport]

ID:
Day: 13 / 11 / 2009 *
Destination: Nice *
Services enabled: Concert[372,350].
Contact:
Transport:

MyTransport found:
- event: France/Italie
- contact: Laurent Vanni
- transport: Voiture [3 place

Figure 9. myTransport on the Nokia N800 Internet tablet



Search results for Related asset =

5 results

Title	Creator	Digital object owner	Archival date	MIME Format
Na Drini ćuprija	Ivo Andrić	Serbian National Heritage	21/02/2010	Text
Miroslavljevo jevanđelje	anonymous	Serbian National Heritage	21/02/2010	JPEG
I prva rukovet	Stevan St. Mokranjac	Serbian National Heritage	21/02/2010	Video
Banjica Concentration Camp	Branka Prpa	Historical Archives of Belg	21/02/2010	HTML
Vasilije Aritmetika	Damjanović	National Center of Digitiz	21/02/2010	PDF

Search results for Related asset = I prva rukovet

Title:	I prva rukovet
Creator:	Stevan St. Mokranjac
Location:	http://www.youtube.com/watch?v=6ca4uvWBDpE
Owner:	Serbian National Heritage Library
Related asset:	
Archivial date:	February 2010, 21
MIME type:	Video
Additional Notes:	Title: First garland

Figure 10. myDistributed Catalog

4.16. myDistributed Catalog for Digitized Cultural Heritage

Participants: Vincenzo Ciancaglini [contact], Bojan Marinkovic [MISANU, Serbia], Liquori Liquori.

Peer-to-peer networks have emerged recently as a flexible decentralized solution to handle large amount of data without the use of high-end servers. We have implemented a distributed catalog built up on an overlay network called "Synapse". The Synapse protocol allows interconnection of different overlay networks each of them being an abstraction of a "community" of virtual providers. Data storage and data retrieval from different kind of content providers (i.e. libraries, archives, museums, universities, research centers, etc.) can be stored inside one catalog. We illustrate the concept based on the Synapse protocol: a catalog for digitized cultural heritage of Serbia, see Figure 10 .

4.17. myStreaming P2P

Participants: Vincenzo Ciancaglini [contact], Rossella Fortuna [Politech Bari], Salvatore Spoto [Univ. Turin], Liquori Liquori, Luigi Alfredo Grieco [Politech Bari].

We have implemented, in Python, a fork of Goalbit http://goalbit.sourceforge.net, an open source video streaming platform peer-to-peer software streaming platform capable of distributing high-bandwidth live video content to everyone preserving its quality. We have aligned with the classical gossip-based distribution protocol a *DHT* that distribute contents according to a content-based strategy.

MYRIADS Project-Team

5. Software

5.1. SAFDIS

Contact: Jean-Louis Pazat, `Jean-Louis.Pazat@irisa.fr`

URL: http://www.irisa.fr/myriads/software/folder.2011-12-13.8949308917/

Status: Version 1.0

License: TBD

Presentation: SAFDIS (Self Adaptation for Distributed Services) is a generic framework allowing the self-adaptation of distributed service based applications within a highly volatile context. Compared to other adaptation frameworks, the main advantages of SAFDIS are its genericity, its distributed nature and the focus on SOAs. SAFDIS is in its final implementation and testing phase within the Myriads team and is being used with a real life use case for emergency services.

The current implementation of SAFDIS is based on a Java OSGi implementation. SAFDIS is written in Java and organized into OSGi bundles. SAFDIS is not tight to any specific operating system and work within any JAVA 1.6 platform. An OSGi implementation is needed (such as the Apache Felix http://felix.apache.org or Equinox eclipse.org/equinox implementations). In order to benefit from the reactive adaptation tools, the Jess engine is also needed as an OSGi bundle (http://www.jessrules.com).

Active contributors (from Myriads project-team): Erwan Daubert, Guillaume Gauvrit, Jean-Louis Pazat.

5.2. HOCL-tools

Contact: Cédric Tedeschi, `Cedric.Tedeschi@irisa.fr`

Status: Version 1.0 to be released

License: TBD

Presentation: HOCL (Higher Order Chemical Language) is a chemical programming language based on the chemical metaphor presented before (see Section 3.5). It was developed for several years within the PARIS team. Within HOCL, following the chemical metaphor, computations can be regarded as chemical reactions, and data can be seen as molecules which participate in these reactions. If a certain condition is held, the reaction will be triggered, thus continuing until it gets inert: no more data can satisfy any computing conditions. To realize this program paradigm, a multiset is implemented to act as a chemical tank, containing necessary data and rules. An HOCL program is then composed of two parts: *chemical rule definitions* (reaction rules) and *multiset definition* (data). More specifically, HOCL provides the high order: reaction rules are molecules that can be manipulated like any other molecules. In other words, HOCL programs can manipulate other HOCL programs.

An HOCL compiler was developed using java to execute some chemical programs expressed with HOCL. This compiler is based on the translation of HOCL programs to java code. As a support for service coordination and service adaptation (refer to Section 6.3), we recently extended the HOCL compiler with the support of decentralized workflow execution. Works around the implementation of a distributed multiset gave birth to an underlying layer for this compiler, making it able to deploy HOCL programs transparently over large scale platforms. This last part is currently considered to be interfaced with the current HOCL compiler. All these features are planned to be released under the common name of *HOCL-tools*.

Active contributors (from Myriads project-team): Héctor Fernández, Marko Obrovac, Cédric Tedeschi.

Impact: The compiler is used as a tool within the team to develop HOCL programs. The decentralized workflow execution support has been used extensively to produce results published and presented at several conferences.

5.3. XtreemOS

Contact: Yvon Jégou, `Yvon.Jegou@inria.fr`

URL: http://www.xtreemos.eu, http://gforge.inria.fr/projects/xtreemos

Status: Version 3.0

License: GPL-2/BSD depending on software packages composing the system

Presentation: XTREEMOS is a Grid Operating system based on Linux with native support for virtual organizations. Three flavours of XTREEMOS were developed for individual PCs, clusters and mobile devices (PDA, notebooks and smartphones). XTREEMOS has been developed by the XTREEMOS consortium.

XTREEMOS software is a set of services developed in Java, C++ and C. XTREEMOS cluster version leverages KERRIGHED single system image operating system. A permanent testbed composed of computers provided by several XTREEMOS partners has been public since fall 2010. The third public version of XTREEMOS has been released in February 2012 for the OpenSuse Linux distribution. Ready-to-use XtreemOS virtual machine images have been made available for the community.

Active contributors (from Myriads project-team): Amine Belhaj, Rémy Garrigue, Yvon Jégou, Christine Morin, Yann Radenac.

Impact: XtreemOS software has been used as part of the COOP ANR project. It was also used in the ANR CLOUD project.Some services such as XtreemFS are used in various R&D projects including Contrail European project.

5.4. Contrail Virtual Execution Platform (VEP)

Contact: Yvon Jégou, `Yvon.Jegou@inria.fr`

URL: http://vep.gforge.inria.fr/index.php?title=Main_Page

Status: Version 1.0

License: BSD

Presentation: Virtual Execution Platform (VEP)[32] is a Contrail service that sits just above IaaS layer at the service provider end of the Contrail cloud federation. The VEP provides a uniform interface for managing the whole lifecycle of elastic applications on the cloud and hides the details of the IaaS layer to the user. VEP applications are described in OVF (Open Virtualization Format) standard format. Resource usage is controlled by CEE (Constrained Execution Environment) rules which can be derived from SLAs (Service Level Agreement). The VEP integrates a monitoring system where the major events about the application, mainly resource usage, are made available to the user.

The VEP service provides a RESTful interface and can be exploited directly by users on top of the provider IaaS. OpenNebula and OCCI-based IaaS interfaces are currently supported.

Active contributors (from Myriads project-team): Roberto Cascella, Florian Dudouet, Filippo Gaudenzi, Piyush Harsh, Yvon Jégou, Christine Morin.

Impact: VEP is part of Contrail software stack. Several Contrail partners experiment use cases on top of VEP. External users can experiment with it using the open testbed operated by Myriads team.

5.5. Snooze

Contact: Christine Morin, `Christine.Morin@inria.fr`

URL: http://snooze.inria.fr

Status: Version 1.0

License: GPLv2

Presentation: Snooze [25], [26], a novel Infrastructure-as-a-Service (IaaS) cloud management system, which is designed to scale across many thousands of servers and virtual machines (VMs) while being easy to configure, highly available, and energy efficient. For scalability, Snooze performs distributed VM management based on a hierarchical architecture. To support ease of configuration and high availability Snooze implements self-configuring and self-healing features. Finally, for energy efficiency, Snooze integrates a holistic energy management approach via VM resource (i.e. CPU, memory, network) utilization monitoring, underload/overload detection and mitigation, VM consolidation (by implementing a modified version of the Sercon algorithm [69]), and power management to transition idle servers into a power saving mode. Snooze is a highly modular Software. It has been extensively evaluated on the Grid'5000 testbed using realistic applications.

Snooze is fully implemented from scratch in Java and currently comprises approximately 15.000 lines of maintainable abstractions-based code. In order to provide a uniform interface to the underlying hypervisors and support transparent VM monitoring and management, Snooze integrates the *libvirt* virtualization library. Snooze provides a RESTful command line interface (CLI) to support virtual cluster (VC) definitions and management (i.e., start, shutdown, destroy, suspend, etc.) as well hierarchy visualization and exporting in GraphML format.

Active contributors (from Myriads team): Eugen Feller, Christine Morin.

Impact: Snooze has been used by students at LIFL, IRIT in France and LBNL in the US in the framework of internships during the summer 2012. It has also been deployed and experimented at EDF R&D. Finally, we know that it was experimented by external users from academia and industry as we received feed-back from them.

5.6. Resilin

Contact: Christine Morin, `Christine.Morin@inria.fr`

URL: http://resilin.inria.fr

Status: Version 1.0

License: GNU Affero GPL

Presentation: Resilin [51], [31] is an open-source system for creating and managing MapReduce execution platforms over clouds. Resilin is compatible with the Amazon Elastic MapReduce (EMR) API, but it goes beyond Amazon's proprietary EMR solution in allowing users (e.g. companies, scientists) to leverage resources from one or more public and/or private clouds. This enables performing MapReduce computations over a large number of geographically-distributed and diverse resources. Resilin can be deployed across most of the open-source and commercial IaaS cloud management systems (e.g., OpenStack, OpenNebula, Amazon EC2). Once deployed, Resilin takes care of provisioning Hadoop clusters and submitting MapReduce jobs, allowing users to focus on writing their MapReduce applications rather than managing cloud resources. Resilin is implemented in the Python language and uses the Apache Libcloud library to interact with IaaS clouds. Resilin has been evaluated on multiple clusters of the Grid'5000 experimentation testbed. The results show that Resilin enables the use of geographically distributed resources with a limited impact on MapReduce job execution time.

Active contributors (from the Myriads project-team): Ancuta Iordache, Nikos Parlavantzas, Christine Morin.

Impact: Resilin is being used in the MOAIS project-team at Inria Grenoble - Rhône Alpes.

5.7. QU4DS

Contact: Jean-Louis Pazat, `Jean-Louis.Pazat@inria.fr`

URL: http://www.irisa.fr/myriads/software/quads/

Status: Version 0.1

License: TBD

Presentation: The QU4DS framework provides PaaS (Platform-as-a-Service) support that fills the gap between the higher-level SaaS (Software-as-a-Service) and the underlying IaaS (Infrastructure-as-a-Service). Qu4DS aids service administrators to define high-level objectives that guide execution management in an automatic and transparent fashion. Moreover, Qu4DS supports the full SLA life-cycle while increasing the service provider profit. SLA support includes service negotiation, instantiation and management on the infrastructure. Orthogonally to these features, complementary actions are in charge of increasing the provider profit guided by SLA constraints.

Currently, Qu4DS targets the development of service providers that use the Master/Worker pattern. Qu4DS assists the development of such services by freeing developers from managing workers and by ensuring their proper execution in accordance with time constraints and by reacting to job failures and delays at runtime. At development time, service developers use the Qu4DS library to develop applications and create a Java jar file. The Qu4DS framework uses this jar file to deploy and manage the service instance on the infrastructure according to SLA constraints.

Active contributors (from Myriads team): André Lage Freitas, Nikos Parlavantzas, Jean-Louis Pazat

5.8. Themis

Contact: Nikos Parlavantzas, `Nikos.Parlavantzas@irisa.fr`

URL: http://www.irisa.fr/myriads/software/themis/

Status: Version 1.0

License: TBD

Presentation: Themis is a market-based private PaaS (Platform-as-a-Service) system, supporting dynamic, fine-grained resource allocation and automatic application management[19], [20]. Themis implements a proportional-share auction that ensures maximum resource utilization while providing incentives to applications to regulate their resource usage. Themis includes generic mechanisms for application deployment and automatic scaling. These mechanisms can be adapted to support diverse performance goals and application types, such as master-worker, MPI, or MapReduce applications. Themis is implemented in Python and uses OpenNebula for virtual machine management. Experimental results on the Grid'5000 testbed show that using Themis increases resource utilization and improves application performance. Themis is currently installed and being evaluated by EDF R&D using EDF high-performance applications.

Active contributors (from the Myriads team): Stefania Costache, Nikos Parlavantzas, Christine Morin.

Impact: Themis is not yet distributed in open source. However, it has been integrated in EDF R&D portal providing access to internal computing resources and is currently experimented on a testbed at EDF R&D.

OASIS Project-Team

5. Software

5.1. ProActive

Participants: F. Baude, D. Caromel, L. Henrio, F. Huet [correspondant], F. Viale, O. Smirnov, B. Sauvan, A. Bourdin.

Proactive Parallel Suite
ProActive is a Java library (Source code under AGPL license) for parallel, distributed, and concurrent computing, also featuring mobility and security in a uniform framework. With a reduced set of simple primitives, ProActive provides a comprehensive API to simplify the programming of applications that are distributed on a Local Area Network (LAN), on cluster of workstations, Clouds, or on Internet Grids.

The library is based on an Active Object pattern that is a uniform way to encapsulate:

- a remotely accessible object,
- a thread,
- an actor with its own script,
- a server of incoming requests,
- a mobile and potentially secure agent.

and has an architecture to inter-operate with (de facto) standards such as:

- Web Service exportation (Apache Axis2 and CXF),
- HTTP transport,
- ssh, rsh, RMI/ssh tunnelling,
- Globus: GT2, GT3, GT4, gsi, Unicore, ARC (NorduGrid)
- LSF, PBS, Sun Grid Engine, OAR, Load Leveler

ProActive is only made of standard Java classes, and requires **no changes to the Java Virtual Machine**, no preprocessing or compiler modification; programmers write standard Java code. Based on a simple Meta-Object Protocol, the library is itself extensible, making the system open for adaptations and optimisations. ProActive currently uses the RMI Java standard library as default portable transport layer, but others such as Ibis or HTTP can be used instead, in an adaptive way.

ProActive is particularly well-adapted for the development of applications distributed over the Internet, thanks to reuse of sequential code, through polymorphism, automatic future-based synchronisations, migration of activities from one virtual machine to another. The underlying programming model is thus innovative compared to, for instance, the well established MPI programming model.

In order to cope with the requirements of large-scale distributed and heterogeneous systems like the Grid, many features have been incorporated into ProActive, including support for many transport and job submission protocols, GCM component support, graphical visualization interface, object migration, distributed and non-functional exception handling, fault-tolerance and checkpointing mechanisms; file transfer capabilities, a job scheduler, a resource manager able to manage various hosting machines, support for JMX and OSGi capabilities, web service object exposition, an SCA personality, etc.

ProActive is a project of the former ObjectWeb, now OW2 Consortium. OW2 is an international consortium fostering the development of open-source middleware for cutting-edge applications: EAI, e-business, clustering, grid computing, managed services and more. For more information, refer to [5] [55] and to the web pages http://www.objectweb.org and http://proactive.inria.fr/ which list several white papers.

ProActive management, distribution, support, and commercialisation is now ensured by the start-up company ActiveEon (http://www.activeeon.com), in the context of a collaboration with Inria and UNS.

5.2. Vercors platform

Participants: E. Madelaine, L. Henrio, A. Savu, M. Alexe.

The Vercors tools (http://www-sop.inria.fr/oasis/Vercors) include front-ends for specifying the architecture and behaviour of components in the form of UML diagrams. We translate these high-level specifications, into behavioural models in various formats, and we also transform these models using abstractions. In a final step, abstract models are translated into the input format for various verification toolsets. Currently we mainly use the various analysis modules of the CADP toolset.

- We have pursued last year experiments in distributed model-checking, and were able to generate explicit state-spaces of (sub-systems) for a new distributed use-case of several billion states. The challenges here lie in the structure of the verification workflow, and in finding strategies for separating the sub-systems in an intelligent way.
- We have also conducted intensive experiments within the Papyrus environment, aiming at the definition of a graphical specification environment combining some of the standard UML formalisms (typically class diagrams and state-machines), with a dedicated graphical formalism for the architecture of GCM components.

5.3. Open Simulation Architecture (OSA)

Participants: O. Dalle, V.D. Nguyen.

OSA stands for Open Simulation Architecture. OSA is primarily intended to be a federating platform for the simulation community: it is designed to favor the integration of new or existing contributions at every level of its architecture. The platform core supports discrete-event simulation engine(s) built on top of the ObjectWeb Consortium's Fractal component model. In OSA, the systems to be simulated are modeled and instrumented using Fractal components. In OSA, the event handling is mostly hidden in the controller part of the components, which alleviates noticeably the modeling process, but also eases the replacement of any part of the simulation engine. Apart the simulation engine, OSA aims at integrating useful tools for modeling, developing, experimenting, and analysing simulations. OSA is also a platform for experimenting new techniques and approaches in simulation, such as aspect oriented programming, separation of concerns, innovative component architectures, and so on.

5.4. BtrPlace

Participant: F. Hermenier.

Btrplace (http://btrp.inria.fr) is an open source virtual machine (VM) placement algorithm for datacenters. BtrPlace has been designed to be extensible. It can be customized by plugins from third party developers to address new SLAs or optimization objectives. Its extensibility is possible thanks to a composable core reconfiguration algorithm implemented using Constraint Programming.

Btrplace is a part of the OW2 project Entropy. It has been originally developed by Fabien Hermenier during its PhD at the Ecole des Mines of Nantes. BtrPlace is now a standalone project that is currently used to address fault tolerance, isolation, infrastructure management, performance, and energy efficiency concerns inside the national project OpenCloudWare (http://opencloudware.org/) and the European project Fit4Green.

This year, our development has been guided by our collaborations. The Fit4Green project chose to rely on BtrPlace to compute an energy-efficient placement for their VMs while some partners inside OpenCloudWare required new placement constraints. The inferring capabilities of BtrPlace and its catalog of placement constraint have then been upgraded accordingly.

PHOENIX Project-Team

5. Software

5.1. DiaSuite: a Development Environment for Sense/Compute/Control Applications

Participants: Charles Consel [correspondent], Julien Bruneau, Amélie Marzin, Damien Martin-Guillerez, Emilie Balland.

Despite much progress, developing a pervasive computing application remains a challenge because of a lack of conceptual frameworks and supporting tools. This challenge involves coping with heterogeneous devices, overcoming the intricacies of distributed systems technologies, working out an architecture for the application, encoding it in a program, writing specific code to test the application, and finally deploying it.

DIASUITE is a suite of tools covering the development life-cycle of a pervasive computing application:

- *Defining an application area.* First, an expert defines a catalog of entities, whether hardware or software, that are specific to a target area. These entities serve as building blocks to develop applications in this area. They are gathered in a taxonomy definition, written in the taxonomy layer of the DIASPEC language.
- *Designing an application.* Given a taxonomy, the architect can design and structure applications. To do so, the DIASPEC language provides an application design layer [40]. This layer is dedicated to an architectural pattern commonly used in the pervasive computing domain [30]. Describing the architecture application allows to further model a pervasive computing system, making explicit its functional decomposition.
- *Implementing an application.* We leverage the taxonomy definition and the architecture description to provide dedicated support to both the entity and the application developers. This support takes the form of a Java programming framework, generated by the DIAGEN compiler. The generated programming framework precisely guides the developer with respect to the taxonomy definition and the architecture description. It consists of high-level operations to discover entities and interact with both entities and application components. In doing so, it abstracts away from the underlying distributed technologies, providing further separation of concerns.
- *Testing an application.*DIAGEN generates a simulation support to test pervasive computing applications before their actual deployment. An application is simulated in the DIASIM tool, without requiring any code modification. DIASIM provides an editor to define simulation scenarios and a 2D-renderer to monitor the simulated application. Furthermore, simulated and actual entities can be mixed. This hybrid simulation enables an application to migrate incrementally to an actual environment.
- *Deploying a system.* Finally, the system administrator deploys the pervasive computing system. To this end, a distributed systems technology is selected. We have developed a back-end that currently targets the following technologies: Web Services, RMI, SIP and OSGI. This targeting is transparent for the application code. The variety of these target technologies demonstrates that our development approach separates concerns into well-defined layers.

This development cycle is summarized in the Figure 1 .

See also the web page http://diasuite.inria.fr.

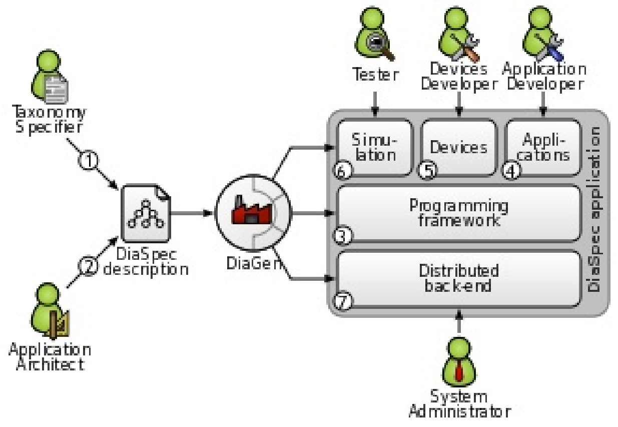


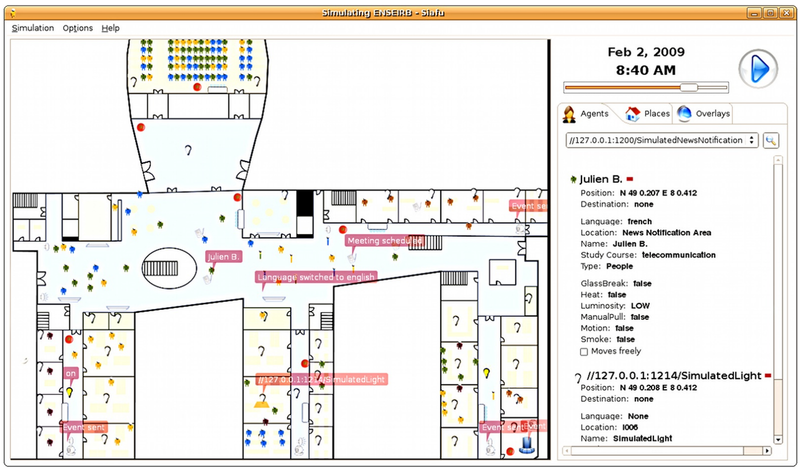
Figure 1. DIASUITE *Development Cycle*

5.1.1. DiaSpec: a Domain-Specific Language for Networked Entities

The core of the DIASUITE development environment is the domain specific language called DIASPEC and its compiler DIAGEN:

- DIASPEC is composed of two layers:
  - The *Taxonomy Layer* allows the declaration of entities that are relevant to the target application area. An entity consists of sensing capabilities, producing data, and actuating capabilities, providing actions. Accordingly, an entity description declares a data source for each one of its sensing capabilities. As well, an actuating capability corresponds to a set of method declarations. An entity declaration also includes attributes, characterizing properties of entity instances. Entity declarations are organized hierarchically allowing entity classes to inherit attributes, sources and actions. A taxonomy allows separation of concerns in that the expert can focus on the concerns of cataloging area-specific entities. The entity developer is concerned about mapping a taxonomical description into an actual entity, and the application developer concentrates on the application logic.
  - The *Architecture Layer* is based on an architectural pattern commonly used in the pervasive computing domain [30]. It consists of context components fueled by sensing entities. These components process gathered data to make them amenable to the application needs. Context data are then passed to controller components that trigger actions on entities. Using an architecture description enables the key components of an application to be identified, allowing their implementation to evolve with the requirements (*e.g.,* varying light management implementations in a controller component to optimize energy consumption).
- DIAGEN is the DIASPEC compiler that performs both static and runtime verifications over DIASPEC declarations and produces a dedicated programming framework that guides and eases the implementation of components. The generated framework is independent of the underlying distributed technology. As of today, DIAGEN supports multiple targets: Local, RMI, SIP, Web Services and OSGI.

5.1.2. DiaSim: a Parametrized Simulator for Pervasive Computing Applications



Feb 2, 2009
8:40 AM

//127.0.0.1:1200/SimulatedNewsNotification

Julien B.
Position: N 49 0.207 E 8 0.412
Destination: none

Language: french
Location: News Notification Area
Name: Julien B.
Study Course: telecommunication
Type: People

GlassBreak: false
Heat: false
Luminosity: LOW
ManualPull: false
Motion: false
Smoke: false
☐ Moves freely

//127.0.0.1:1214/SimulatedLight
Position: N 49 0.208 E 8 0.412
Destination: none

Language: None
Location: I006
Name: SimulatedLight

Figure 2. A screenshot of the DIASIM *simulator*

Pervasive computing applications involve both software and integration concerns. This situation is problematic for testing pervasive computing applications because it requires acquiring, testing and interfacing a variety of software and hardware entities. This process can rapidly become costly and time-consuming when the target environment involves many entities.

To ease the testing of pervasive applications, we are developing a simulator for pervasive computing applications: DIASIM. To cope with widely heterogeneous entities, DIASIM is parameterized with respect to a DIASPEC specification describing a target pervasive computing environment. This description is used to generate with DIAGEN both a programming framework to develop the simulation logic and an emulation layer to execute applications. Furthermore, a simulation renderer is coupled to DIASIM to allow a simulated pervasive system to be visually monitored and debugged. The simulation renderer is illustrated in Figure 2 .

5.2. DiaSuiteBox: an Open Service Platform

Participants: Julien Bruneau [correspondent], Damien Martin-Guillerez, Charles Consel, Emilie Balland.

The DiaSuiteBox platform runs an open-ended set of applications leveraging a range of appliances and web services. Our solution consists of a dedicated development environment, a certifying application store, and a lightweight runtime platform. This solution is based on the DIASUITE project.

The DiaSuiteBox platform can be embedded in a small plug-computer or deployed in the cloud. Thanks to the application store and the developer community, the platform is fed by a full offer of new innovative applications. During the submission process, an application is automatically analyzed and checked in order to be certified. The user is ensured of the behavior of its applications are innocuous and correct beside the provided information. This box relies on several technology standards like UPnP, Bluetooth, USB, etc. As shown in Figure 3 , this platform can be easily extended by plugging appliances directly on the box or by connecting devices on the local network.

See also the web page http://diasuitebox.inria.fr.

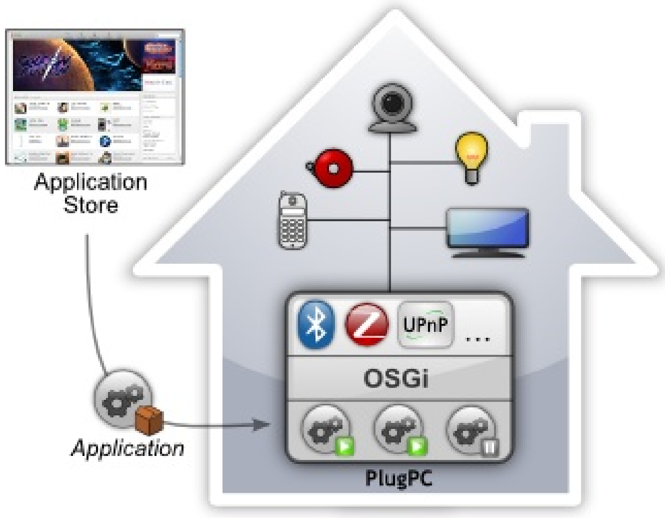


Figure 3. DiaSuiteBox platform architecture

REGAL Project-Team

5. Software

5.1. Coccinelle

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

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 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.

http://coccinelle.lip6.fr

5.2. SwiftCloud

Participants: Marc Shapiro [correspondent], Marek Zawirski, Annette Bieniusa, Valter Balegas.

SwiftCloud is a platform for deploying large-scale distributed applications on the edge, close to the users. Internet delays are a problem for interactive distributed applications. Truly optimal responsiveness and availability require mutable shared state replicated near the client, at the network edge. This raises serious challenges of consistency, fault tolerance, and programmability. The SwiftCloud system is designed to address these challenges. Our data model is based on client-side caching of shared mutable objects, made practical with a library of synchronisation-free, yet provably correct object types (CRDTs). The consistency model combines eventual consistency, absence of roll-backs, transactional consistency and session guarantees, but stops short of serialisability. This programming model is practically useful, yet does not require synchronisation, thus ensuring scalability. Maintaining the guarantees at a reasonable cost is especially challenging at large scale. Scalability and programmability are helped by several design decisions detailed in the paper. We validated our approach by building the SwiftCloud platform, by deploying three significant applications, and by measuring their performance in different configurations, in order to explore the benefits of replication at different locations.

SwiftCloud was supported by the ConcoRDanT ANR project (Section 7.1.4) and a Google European Doctoral Fellowship (Section 7.2.2.1).

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

5.3. Treedoc

Participants: Marc Shapiro [correspondent], Marek Zawirski, Nuno Preguiça.

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.

TreeDoc is supported by the Prose, Streams and ConcoRDanT ANR projects (Sections 7.1.7 , 7.1.6 and 7.1.4 respectively) and by a Google European Doctoral Fellowship (Section 7.2.2.1).

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

5.4. 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.

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

5.5. Java and .Net runtimes for LLVM

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

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 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 its 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://vmkit2.gforge.inria.fr/

RMOD Project-Team

4. Software

4.1. Moose

Participants: Stéphane Ducasse [correspondant], Muhammad Bhatti, Andre Hora, Nicolas Anquetil, Tudor Gîrba [University of Bern].

Web: http://www.moosetechnology.org/

The platform. Moose is a language-independent environment for reverse- and re-engineering complex software systems. Moose provides a set of services including a common meta-model, metrics evaluation and visualization, a model repository, and generic GUI support for querying, browsing and grouping. The development of Moose began at the Software Composition Group in 1997, and is currently contributed to and used by researchers in at least seven European universities. Moose offers an extensible meta-described metamodel, a query engine, a metric engine and several visualizations. Moose is currently in its fourth major release and comprises 55,000 lines of code in 700 classes.

The RMoD team is currently the main maintainer of the Moose platform. There are 200 publications (journal, international conferences, PhD theses) based on execution or use of the Moose environment.

The first version running on top of Pharo (Moose 4.0) was released in June 2010. In February 2012, Moose 4.6 was released.
Here is the self-assessment of the team effort following the grid given at http://www.inria.fr/institut/organisation/instances/commission-d-evaluation.

- **(A5)** Audience : 5 – Moose is used by several research groups, a consulting company, and some companies using it in ad-hoc ways.
- **(SO4)** Software originality : 4 – Moose aggregates the last results of several research groups.
- **(SM4)** Software Maturity : 4 – Moose is developed since 1996 and got two main redesign phases.
- **(EM4)** Evolution and Maintenance : 4 – Moose will be used as a foundation of our Synectique start up so its maintenance is planned.
- **(SDL4)** Software Distribution and Licensing : 4 – Moose is licensed under BSD
- **(OC)** Own Contribution : (Design/Architecture)DA-4, (Coding/Debugging)-4, (Maintenance/Support)-4, (Team/Project Management)-4

4.2. Pharo

Participants: Marcus Denker [correspondant], Damien Cassou, Stéphane Ducasse, Esteban Lorenzano, Mariano Martinez-Peck, Damien Pollet, Igor Stasenko, Veronica Uquillas-Gomez.

Web: http://www.pharo-project.org/

The platform. Pharo is a new open-source Smalltalk-inspired language and environment. It provides a platform for innovative development both in industry and research. By providing a stable and small core system, excellent developer tools, and maintained releases, Pharo's goal is to be a platform to build and deploy mission critical Smalltalk applications.

The first stable version, Pharo 1.0, was released in 2010. The development of Pharo accelerated in 2011 and 2012: Versions 1.2 to 1.4 have been released (with more than 2400 closed issues), and the development branch (2.0) has seen already over 398 incremental releases as of mid November 2012. In 2012, RMoD organized the first *Pharo Conference* during two days in May with 60 participants.

Additionally, in November 2012 RMoD launched the Pharo Consortium (http://www.pharo-project.org/community/consortium) and the Pharo Association (http://association.pharo.org/). 25 companies already shown interest in supporting the consortium.

RMoD is the main maintainer and coordinator of Pharo.
Here is the self-assessment of the team effort following the grid given at http://www.inria.fr/institut/organisation/instances/commission-d-evaluation.

- **(A5)** Audience: 5 – Used in many universities for teaching, more than 25 companies.
- **(SO3)** Software originality : 3 – Pharo offers a classical basis for some aspects (UI). It includes new frameworks and concepts compared to other Smalltalk implementations.
- **(SM4)** Software Maturity: 4 – Bug tracker, continuous integration, large test suites are on place.
- **(EM4)** Evolution and Maintenance: 4 – Active user group, consortium and association had just been set up.
- **(SDL4)** Software Distribution and Licensing: 4 – Pharo is licensed under MIT.
- **(OC5)** Own Contribution: (Design/Architecture) DA-5, (Coding/Debugging) CD-5, (Maintenance/Support) MS-5, (Team/Project Management) TPM-5

4.3. Fuel

Participants: Martin Dias [Correspondant], Mariano Martinez-Peck.

Web: http://rmod.lille.inria.fr/web/pier/software/fuel

Objects in a running environment are constantly being born, mutating their status and dying in the volatile memory of the system. The goal of serializers is to store and load objects either in the original environment or in another one. Fuel is a general-purpose serializer based on four principles: (1) speed, through a compact binary format and a pickling algorithm which obtains the best performance on materialization; (2) good object-oriented design, without any special help from the virtual machine; (3) specialized for Pharo, so that core objects (such as contexts, block closures and classes) can be serialized too; (4) flexible about how to serialize each object, so that objects are serialized differently depending on the context.
Here is the self-assessment of the team effort following the grid given at http://www.inria.fr/institut/organisation/instances/commission-d-evaluation.

- **(A4)** Audience: 4 – Large audience software, usable by people inside and outside the field with a clear and strong dissemination, validation, and support action plan.
- **(SO3)** Software originality : 3.
- **(SM4)** Software Maturity: 4 – Bug tracker, continuous integration, large test suites are on place.
- **(EM4)** Evolution and Maintenance: 4.
- **(SDL4)** Software Distribution and Licensing: 4 – Fuel is licensed under MIT.
- **(OC5)** Own Contribution: (Design/Architecture) DA-5, (Coding/Debugging) CD-5, (Maintenance/Support) MS-5, (Team/Project Management) TPM-5

4.4. Athens

Participant: Igor Stasenko [Correspondant].

Athens is a vector graphics framework for Pharo.

4.5. Citezen

Participants: Damien Pollet [Correspondant], Stéphane Ducasse.

Web: http://people.untyped.org/damien.pollet/software/citezen/

Citezen is a suite of tools for parsing, validating, sorting and displaying BibTeX databases. This tool suite is integrated within the Pier Content Management System (CMS) and both are implemented on top of Pharo. Citezen aims at replacing and extending BibTeX, in Smalltalk; ideally, features would be similar to BibTeX, CrossTeX, and CSL.

4.6. Handles

Participant: Jean-Baptiste Arnaud [Correspondant].

Web: http://jeanbaptiste-arnaud.eu/handles/

An Handle is a first-class reference to a target object. Handles can alter the behavior and isolate the state of the target object. Handles provide infrastructure to automatically create and wrap new handles when required. A real-time control of handles is possible using a special object called metaHandle.

4.7. Hazelnut

Participants: Guillermo Polito [Correspondant], Benjamin van Ryseghem, Nicolas Paez, Igor Stasenko.

Web: http://rmod.lille.inria.fr/web/pier/software/Seed

Traditionally, Smalltalk-based systems are not bootstrapped because of their ability to evolve by self-modification. Nevertheless, the absence of a bootstrap process exposes many problems in these systems, such as the lack of reproducibility and the impossibility to reach certain evolution paths. Hazelnut is a tool that aims to introduce a bootstrap process into these systems, in particular Pharo.

4.8. Jet

Participant: Veronica Uquillas-Gomez [Correspondant].

Jet is a tool to analyze streams of changes. Jet identifies dependencies between changes and sets of changes and supports cherry picking. Moreover, Jet classifies sets of changes based on their dependencies as a way to ease the analysis of changes within the stream and guide system integrators.

4.9. LegacyParsers

Participants: Muhammad Bhatti [Correspondant], Nicolas Anquetil, Guillaume Larcheveque, Esteban Lorenzano, Gogui Ndong.

As part of our research on legacy software and also for the Synectique company), we started to define several parsers for old languages like Cobol for example. This work is important to help us validate our meta-model and tools against a larger range of existing technologies and to discover the limits of our approach. From our initial results, and the in-depth understanding that it gave us, we are formulating new research objectives in meta-model driven reverse engineering. This work is also important for the spin-off company, as being able to work with such technologies is fundamental.

4.10. Mate

Participants: Marcus Denker [Correspondant], Clement Bera, Camillo Bruni.

Mate is the future research-oriented virtual machine for Pharo. Its goal is to serve as a prototype for researchers to experiment with. As a result, the design of Mate is very simple to understand. As of today, Mate consists of an AST interpreter, a new object memory layout, and a simple garbage collector.

4.11. NativeBoost

Participant: Igor Stasenko [Correspondant].

Web: http://code.google.com/p/nativeboost/

NativeBoost is a Smalltalk framework for generating and running machine code from the language side of Pharo. As part of it comes a foreign function interface that enables calling external C functions from Smalltalk code with minimal effort.

4.12. Nabujito

Participants: Camillo Bruni [Correspondant], Marcus Denker.

Nabujito is a new Just In Time compiler implemented as a Smalltalk application, based on NativeBoost, that does not require changes in the virtual machine.

4.13. Nautilus

Participants: Benjamin Van Ryseghem [Correspondant], Stéphane Ducasse, Igor Stasenko, Camillo Bruni, Esteban Lorenzano.

Nautilus is a new source code browser based on the latest infrastructure representations. Its goal is mainly to replace the current system browser that was implemented in the 80s and that doesn't provide optimal tools for the system as it has evolved.

4.14. SourceCity

Participants: Erwan Douaille [Correspondant], Igor Stasenko, Guillaume Larcheveque, Stéphane Ducasse.

Modern systems are too complex. Understanding and analyzing these systems is very hard and tedious (thousand of classes, millions of lines of code). One needs an overview of the system that allows to discover important parts in the system, weak points, suspicious components. SourceCity is a powerful 3D tool that can help to understand quickly how a system works by taking the metaphor of a city buildings. By looking at tall, large, low building, one can identify different properties of the software components being represented.

4.15. Spec

Participants: Benjamin Van Ryseghem [Correspondant], Stéphane Ducasse, Johan Fabry.

Spec is a programming framework for generating graphical user interfaces inspired by VisualWorks' Subcanvas. The goal of Spec is to tackle the lack of reuse experienced in existing tools. Spec serves as a pluggable layer on top of multiple lower-level graphical frameworks. Many improvements have been noticed in Pharo after the introduction of Spec in terms of speed or number of lines of code while we re-implemented existing tools using Spec.

4.16. VerveineJ

Participants: Nicolas Anquetil [Correspondant], Andre Hora, Guillaume Larcheveque.

Web: Inria project https://gforge.inria.fr/projects/verveinej/.

VerveineJ is a tool to export Java projects into the MSE format, which can then be imported inside Moose (see above). Although VerveineJ is not a research project in itself, it is an important building block for our research in that it allows us to run the Moose platform on legacy Java projects. Another similar tool, Infusion, already existed to fulfil the same needs, but it was closed sources and presented some errors that tainted the results we could obtain.

SARDES Project-Team

4. Software

4.1. AAC_tactics

Participants: Thomas Braibant, Damien Pous [correspondant].

AAC_tactics is a plugin for the Coq proof-assistant that implements new proof tactics for rewriting modulo associativity and commutativity. It is available at http://sardes.inrialpes.fr/~braibant/aac_tactics and as part of the Coq distribution.

- ACM: D.2.4 Software/Program Verification
- Keywords: Rewriting, rewriting modulo AC, proof tactics, proof assistant
- Software benefit: AAC_tactics provides novel efficient proof tactics for rewriting modulo associativity and commutativity.
- License: LGPL
- Type of human computer interaction: N/A
- OS/Middleware: Windows, Linux, MacOS X
- Programming language: Coq

4.2. ATBR

Participants: Thomas Braibant, Damien Pous [correspondant].

ATBR (Algebraic Tools for Binary Relations) is library for the Coq proof assistant that implements new proof tactics for reasoning with binary relations. Its main tactics implements a decision procedure for inequalities in Kleene algebras. It is available at http://sardes.inrialpes.fr/~braibant/atbr and as part of the Coq distribution contributed modules.

- ACM: D.2.4 Software/Program Verification
- Keywords: Binary relations, Kleene algebras, proof tactics, proof assistant
- Software benefit: ATBR provides new proof tactics for reasoning with binary relations.
- License: LGPL
- Type of human computer interaction: N/A
- OS/Middleware: Windows, Linux, MacOS X
- Programming language: Coq

4.3. MoKa

Participant: Sara Bouchenak [correspondant].

MoKa is a software framework for the modeling and capacity planning of distributed systems. It first provides a set of tools to build analytical models that describe the behavior of distributed computing systems, in terms of performance, availability, cost. The framework allows to include several model algorithms and to compare them regarding their accuracy and their efficiency. Furthermore, MoKa provides a set of tools to build capacity planning methods. A capacity planning method allows to find a distributed system configuration that guarantee given quality-of-service objectives. MoKa is able to include different capacity planning algorithms and to compare them regarding their efficiency and the optimality of their results. MoKais available at: http://sardes.inrialpes.fr/research/moka.

- ACM: C.2.4 Distributed Systems, C.4 Performance of Systems, D.2.9 Management
- Keywords: Caching, multi-tier systems, consistency, performance
- Software benefit: a novel end-to-end caching protocol for multi-tier services.
- License: TBD
- Type of human computer interaction: command-line interface
- OS/Middleware: Windows, Linux, MacOS X
- Programming language: Java

4.4. ConSer

Participant: Sara Bouchenak [correspondant].

CONSER is a software framework for the modeling and the concurrency and admision control of servers systems. It implements a fluid-based model that exhibits the dynamics and behavior of a server system in terms of service performance and availability. CONSER implements various novel admission control laws for servers such as $AM\text{-}C$, $PM\text{-}C$, $AA\text{-}PM\text{-}C$ and $PA\text{-}AM\text{-}C$. A control law produces the server concurrency level that allows to trade-off and meet given service level objectives. CONSER's modeling and control laws algorithms are implemented following a proxy-based approach for more transparency.

- ACM: C.4 Performance of Systems; D.2.9 Management
- Keywords: System management, capacity planning, performance management
- Software benefit: MoKa provides modeling, capacity planning and performance management facilities for application server clusters. Thanks to its model-based capacity planning, MoKa is able to enforce service level objectives while minimizing the service cost.
- License: LGPL
- Type of human computer interaction: web interface
- OS/Middleware: Windows, Linux, MacOS X
- Programming language: Java, AspectJ

4.5. e-Caching

Participants: Damian Serrano-Garcia, Sara Bouchenak [correspondant].

E-CACHING is a software framework for higher scalability of multi-tier Internet services through end-to-end caching of dynamic data. It provides a novel caching solution that allows to cache different types of data (e.g. Web content, database query results, etc.), at different locations of multi-tier Internet services. The framework allows to combine different caches and, thus, to provide higher scalability of Internet services. E-CACHING maintains the integrity of the cached data through novel distributed caching algorithms that guarantee the consistency of the underlying data.

- ACM: C.2.4 Distributed Systems, C.4 Performance of Systems
- Keywords: Caching, multi-tier systems, consistency, performance
- Software benefit: a novel end-to-end caching protocol for multi-tier services, consistency management, performance improvement.
- License: TBD
- Type of human computer interaction: command-line interface
- OS/Middleware: Windows, Linux, MacOS X
- Programming language: Java

4.6. MRB

Participants: Amit Sangroya, Dàmian Serrano-Garcia, Sara Bouchenak [correspondant].

MRB is a software framework for benchmarking the performance and dependability of MapReduce distributed systems. It includes five benchmarks covering several application domains and a wide range of execution scenarios such as data-intensive vs. compute-intensive applications, or batch applications vs. interactive applications. MRB allows to characterize application workload, faultload and dataload, and it produces extensive performance and dependability statistics.

- ACM: C.2.4 Distributed Systems, C.4 Performance of Systems
- Keywords: Benchmark, performance, dependability, MapReduce, Hadoop, Cloud Computing
- Software benefit: the first performance and dependability benchmark suite for MapReduce systems.
- License: TBD
- Type of human computer interaction: GUI and command-line interface
- OS/Middleware: Windows, Linux, MacOS X
- Programming language: Java, Unix Shell scripts

4.7. BZR

Participants: Eric Rutten [correspondant], Gwenaël Delaval [POP ART team].

BZR is a reactive language, belonging to the synchronous languages family, whose main feature is to include discrete controller synthesis within its compilation. It is equipped with a behavioral contract mechanisms, where assumptions can be described, as well as an enforce property part: the semantics of the latter is that the property should be enforced by controlling the behaviour of the node equipped with the contract. This property will be enforced by an automatically built controller, which will act on free controllable variables given by the programmer.

BZR is now further developed with the Pop-Art team, where G. Delaval got a position. It has been designed and developed in the Sardes team in relation with the research topic on Model-based Control of Adaptive and Reconfigurable Systems. It is currently applied in different directions: component-based design and the Fractal framework; real-time control systems and the Orccad design environment; operating systems and administration loops in virtual machines; hardware and reconfigurable architecture (FPGAs).

See also the web page http://bzr.inria.fr.

- ACM: D.3.3 [Programming Languages]: Language Constructs and Features—Control structures; C.3 [Special-purpose and Application-based Systems]: Real-time and embedded systems; D.2.2 [Software Engineering]: Design Tools and Techniques—Computer-aided software engineering, State diagrams; D.2.4 [Software Engineering]: Software / Program Verification—Formal methods, Programming by contract
- Keywords: Discrete controller synthesis, modularity, components, contracts, reactive systems, synchronous programming, adaptive and reconfigurable systems
- Software benefit: the first integration of discrete control synthesis in a compiler, making it usable at the level of the programming language.
- License: TBD
- Type of human computer interaction: programming language and command-line interface
- OS/Middleware: Linux
- Programming language: Caml; generates C or Java or Caml executable code

SCORE Team

5. Software

5.1. Wiki3.0

Participants: Luc André, Bogdan Flueras, Claudia-Lavinia Ignat [contact], Gérald Oster.

In the context of the Wiki 3.0 project (http://wiki30.xwikisas.com/) (december 2009 - june 2012) sponsored by the call for projects “Innovative Web” launched by the French Ministry of Economy, SCORE team designed and integrated real-time editing features into the XWiki system (http://www.xwiki.org). We designed solutions for a raw text editor as well as for a WYSIWYG editor for XWiki pages. The real-time wiki editor has been released as an extension of XWiki (http://extensions.xwiki.org/xwiki/bin/view/Extension/RealTime+Wiki+Editor).

TRISKELL Project-Team

5. Software

5.1. Kermeta

Participants: Didier Vojtisek [correspondant], Olivier Barais, Arnaud Blouin, Benoit Combemale, Jacques Falcou, François Fouquet, Marie Gouyette, Clément Guy, Jean-Marc Jézéquel, Jonathan Marchand.

Nowadays, object-oriented meta-languages such as MOF (meta-object Facility) are increasingly used to specify domain-specific languages in the model-driven engineering community. However, these meta-languages focus on structural specifications and have no built-in support for specifications of operational semantics. Integrated with the industrial standard Ecore and aligned with the OMG standard EMOF 2.0, the Kermeta language consists in a extension to these meta languages to support behavior definition. The language adds precise action specifications with static type checking and genericity at the meta level. Based on object-orientation and aspect orientation concepts, the Kermeta language adds model specific concepts. It is used in several use cases:

- to give a precise semantic of the behavior of a metamodel which then can be simulated.
- to act as a model transformation language.
- to act as a constraint language.

The development environment built for the Kermeta language provides an integrated workbench based on Eclipse. It offers services such as : model execution, text editor (with syntax higlighting, code autocompletion), additional views and various import/export transformations.

Thanks to Kermeta it is possible to build various frameworks dedicated to domain specific metamodels. Those frameworks are organised into MDKs (Model Development Kits). For example, Triskell proposes MDKs to work with metamodels such as Java5, UML2, RDL (requirements), Ecore, Traceability,...

In 2011, Kermeta tooling has been refactored into a version 2.0.x in order to ease the integration of various MOF related languages in the tool chain. This new version also focuses on a fully compiled mode that allows to deploy Kermeta programs in production environments.

See also the web page http://www.kermeta.org.

- APP: IDDN.FR.001.420009.000.S.P.2005.000.10400
- Version: 2.0.1
- Programming language: Java, Scala, Kermeta

Main competitors:

- XMF-Mosaic is developed by Ceteva and is now open-source since 2008.
- GME is a large scale Meta-Modeling Environment developed at Vanderbilt University (ISIS project) since 2002.
- MOFLON is a Metamodeling Framework with Graph Transformations, developed by A. Schuerr's group (TU-Darmstadt) since 2008.
- XCore is a recent (2011) Eclipse project supported by Itemis/Macro Modelling that provides a single operational surface syntax for Ecore.
- Many QVT inspired model transformation tools focused on model transformations.

Main innovative features:

Kermeta was one of the first solutions to offer an operational semantics on top of EMOF. It still proposes several unique features that cannot be found in the tools presented above, such as:

- aspect weaving at the metamodel level allows fast prototyping of a wide variety of tools;
- model typing allows a safe reuse of algorithms and transformations accross diffrent metamodels.

5.2. Kevoree

Participants: Olivier Barais [correspondant], François Fouquet, Erwan Daubert, Jean-Émile Dartois, Johann Bourcier, Antonio Mattos, Noël Plouzeau.

Kevoree is an open-source models@runtime platform [1] to properly support the dynamic adaptation of distributed systems. Models@runtime basically pushes the idea of reflection [82] one step further by considering the reflection layer as a real model that can be uncoupled from the running architecture (e.g. for reasoning, validation, and simulation purposes) and later automatically resynchronized with its running instance.

Kevoree has been influenced by previous work that we carried out in the DiVA project [82] and the Entimid project [83]. With Kevoree we push our vision of models@runtime [81] farther. In particular, Kevoree provides a proper support for distributed models@runtime. To this aim we introduced the *Node* concept to model the infrastructure topology and the *Group* concept to model semantics of inter node communication during synchronization of the reflection model among nodes. Kevoree includes a *Channel* concept to allow for multiple communication semantics between remote*Components* deployed on heterogeneous nodes. All Kevoree concepts (Component, Channel, Node, Group) obey the object type design pattern to separate deployment artifacts from running artifacts. Kevoree supports multiple kinds of very different execution node technology (e.g. Java, Android, MiniCloud, FreeBSD, Arduino, ...).

Kevoree is distributed under the terms of the LGPL open source license.

Main competitors:

- the Fractal/Frascati eco-system [2].
- SpringSource Dynamic Module [3]
- GCM-Proactive [4]
- OSGi [5]
- Chef [6]
- Vagran [7]

Main innovative features:

- distributed models@runtime platform (with a distributed reflection model and an extensible models@runtime dissemination set of strategies).
- Support for heterogeneous node type (from Cyber Physical System with few resources until cloud computing infrastructure).
- Fully automated provisioning model to correctly deploy software modules and their dependencies.
- Communication and concurrency access between software modules expressed at the model level (not in the module implementation).

Impact: Several European projects leveraging the Kevoree platform have recently been accepted. Besides we are currently developing a testbed named DAUM. This testbed is developed since mid 2011 to experiment with Kevoree in real life situations. More precisely, DAUM is a highly dynamic pervasive system that mixes wireless smart sensors, user interaction devices such as digital pads, and distributed data servers in a cloud. The current specialization of DAUM is a distributed tactical information and decision system for firefighters. This application includes individual sensors in the personal protective equipment of firefighters, embedded computation nodes that are fully reconfigurable in real time and over the air, distributed monitoring servers

[1] http://www.kevoree.org
[2] http://frascati.ow2.org
[3] http://www.springsource.org/osgi/
[4] http://proactive.inria.fr/
[5] http://www.osgi.org
[6] http://wiki.opscode.com/display/chef/Deploy+Resource
[7] http://vagrantup.com/

in trucks, and personal computers for information access and decision making. The DAUM platform is used internally to try research results on distributed *models@runtime*. DAUM is used externally to prepare and support cooperation activities with other research teams (the Myriads Inria team is a partner of DAUM) and with potential industrial partners.

See also the web page http://www.kevoree.org.

- Version: 1.0
- Programming language: Java, Scala, Kermeta

ALGORILLE Project-Team

5. Software

5.1. Introduction

Software is a central part of our output. In the following we present the main tools to which we contribute. We use the Inria software self-assessment catalog for a classification.

5.2. parXXL

Participants: Jens Gustedt, Stéphane Vialle.

ParXXL is a library for large scale computation and communication that executes fine grained algorithms on coarse grained architectures (clusters, grids, mainframes). It is one of the software bases of the InterCell project and has been proven to be a stable support, there. It is available under a GPLv2 at http://parxxl.gforge.inria.fr/. ParXXL is not under active development anymore, but still maintained in the case of bugs or portability problems.
Software classification: A-3, SO-4, SM-3, EM-2, SDL-4, DA-4, CD-4, MS-2, TPM-2

5.3. Distem

Participants: Tomasz Buchert, Emmanuel Jeanvoine, Lucas Nussbaum, Luc Sarzyniec.

Wrekavoc and Distem are distributed system emulators. They enable researchers to evaluate unmodified distributed applications on heterogeneous distributed platforms created from an homogeneous cluster: CPU performance and network characteristics are altered by the emulator.
Wrekavoc was developed until 2010, and we then focused our efforts on **Distem**, that shares the same goals with a different design. Distem is available from http://distem.gforge.inria.fr/ under GPLv3.
Software classification: A-3-up, SO-4, SM-3-up, EM-3, SDL-4, DA-4, CD-4, MS-4, TPM-4.

5.4. SimGrid

Participants: Martin Quinson, Marion Guthmuller, Paul Bédaride, Lucas Nussbaum.

SimGrid is a toolkit for the simulation of distributed applications in heterogeneous distributed environments. The specific goal of the project is to facilitate research in the area of parallel and distributed large scale systems, such as Grids, P2P systems and clouds. Its use cases encompass heuristic evaluation, application prototyping or even real application development and tuning. SimGrid has an active user community of more than one hundred members, and is available under GPLv3 from http://simgrid.gforge.inria.fr/.
Software classification: A-4-up, SO-4, SM-4, EM-4, SDL-5, DA-4, CD-4, MS-3, TPM-4.

5.5. ORWL and P99

Participant: Jens Gustedt.

ORWL is a reference implementation of the Ordered Read-Write Lock tools as described in [4]. The macro definitions and tools for programming in C99 that have been implemented for ORWL have been separated out into a toolbox called P99. ORWL is intended to become opensource, once it will be in a publishable state. P99 is available under a QPL at http://p99.gforge.inria.fr/.
Software classification: A-3-up, SO-4, SM-3, EM-3, SDL (P99: 4, ORWL: 2-up), DA-4, CD-4, MS-3, TPM-4

5.6. Kadeploy

Participants: Luc Sarzyniec, Emmanuel Jeanvoine, Lucas Nussbaum.

Kadeploy is a scalable, efficient and reliable deployment (provisioning) system for clusters and grids. It provides a set of tools for cloning, configuring (post installation) and managing cluster nodes. It can deploy a 300-nodes cluster in a few minutes, without intervention from the system administrator. It plays a key role on the Grid'5000 testbed, where it allows users to reconfigure the software environment on the nodes, and is also used on a dozen of production clusters both inside and outside INRIA. It is available from http://kadeploy3.gforge.inria.fr/ under the Cecill license.
Software classification: A-4-up, SO-3, SM-4, EM-4, SDL-4-up, DA-4, CD-4, MS-4, TPM-4.

AVALON Team

5. Software

5.1. BitDew

Participants: Gilles Fedak [correspondant], Haiwu He, Bing Tang, José Francisco Saray Villamizar, Mircea Moca, Lu Lu.

BITDEW is an open source middleware implementing a set of distributed services for large scale data management on Desktop Grids and Clouds. BITDEW relies on five abstractions to manage the data : i) replication indicates how many occurrences of a data should be available at the same time on the network, ii) fault-tolerance controls the policy in presence of hardware failures, iii) lifetime is an attribute absolute or relative to the existence of other data, which decides of the life cycle of a data in the system, iv) affinity drives movement of data according to dependency rules, v) protocol gives the runtime environment hints about the protocol to distribute the data (http, ftp, or bittorrent). Programmers define for every data these simple criteria, and let the BITDEW runtime environment manage operations of data creation, deletion, movement, replication, and fault-tolerance operation.

BITDEW is distributed open source under the GPLv3 or Cecill licence at the user's choice. 10 releases were produced over the last two years, and it has been downloaded approximately 6,000 times on the Inria forge. Known users are Université Paris-XI, Université Paris-XIII, University of Florida (USA), Cardiff University (UK) and University of Sfax (Tunisia). In terms of support, the development of BitDew is partly funded by the Inria ADT BitDew and by the ANR MapReduce projects. Thanks to this support, we have developed and released the first prototype of the MapReduce programming model for Desktop Grids on top of BitDew. In 2012, 8 versions of the software have been released, including the version 1.2.0 considered as a stable release of BitDew with many advanced features. Our most current work focuses on providing reliable storage on top of hybrid distributed computing infrastructures.

5.2. SBAM

Participants: Eddy Caron [correspondant], Florent Chuffart.

SBAM (http://graal.ens-lyon.fr/SBAM) is the middleware directly coming from results of the ANR project SPADES. SBAM initiates a non-intrusive, but highly dynamic environment able to take advantages of available resources without disturbing their native mechanism. SBAM federates multisite resources in order to schedule, submit and compute users' tasks in a transparent way.

SBAM is, firstly, a decentralized grid middleware. It relies on a P2P approach, i.e., a set of agents able to discover resources and schedule computing tasks over a federation of heterogeneous computing platforms (petascale computers, data centers, clouds, ...). SBAM dynamically acquires and releases resources of computing sites according to users' needs and conditions, to federate them into a global constantly growing or shrinking logical platform, referred to as the overlay.

5.3. DIET

Participants: Daniel Balouek, Eddy Caron [correspondant], Frédéric Desprez, Maurice Djibril Faye, Cristian Klein, Arnaud Lefray, Guillaume Mercier, Adrian Muresan, Jonathan Rouzaud-Cornabas, Lamiel Toch, Huaxi Zhang.

Huge problems can now be processed over the Internet thanks to Grid and Cloud middleware systems. The use of on-the-shelf applications is needed by scientists of other disciplines. Moreover, the computational power and memory needs of such applications may of course not be met by every workstation. Thus, the RPC paradigm seems to be a good candidate to build Problem Solving Environments on the Grid or Cloud. The aim of the DIET project (http://graal.ens-lyon.fr/DIET) is to develop a set of tools to build computational servers accessible through a GridRPC API.

Moreover, the aim of a middleware system such as DIET is to provide a transparent access to a pool of computational servers. DIET focuses on offering such a service at a very large scale. A client which has a problem to solve should be able to obtain a reference to the server that is best suited for it. DIET is designed to take into account the data location when scheduling jobs. Data are kept as long as possible on (or near to) the computational servers in order to minimize transfer times. This kind of optimization is mandatory when performing job scheduling on a wide-area network. DIET is built upon *Server Daemons*. The scheduler is scattered across a hierarchy of *Local Agents* and *Master Agents*. Applications targeted for the DIET platform are now able to exert a degree of control over the scheduling subsystem via *plug-in schedulers*. As the applications that are to be deployed on the Grid vary greatly in terms of performance demands, the DIET plug-in scheduler facility permits the application designer to express application needs and features in order that they be taken into account when application tasks are scheduled. These features are invoked at runtime after a user has submitted a service request to the MA, which broadcasts the request to its agent hierarchy.

In 2012, our objective was to extend DIET to benefit from virtualized resources such as ones coming from cloud platforms. Wa have designed how it can be extended to access virtualized resources. We can easily support new cloud service providers and cloud middleware systems. We have prototyped the new version of DIET which benefits from virtualized resources. As cloud resources are dynamic, we have on-going research in the field of automatic and elastic deployment for middleware systems. DIET will be able to extend and reduce the amount on aggregated resources and adjust itself when resources fail. We have started works to extend our data management software, DAGDA, to take advantage of cloud storage and the new data computing paradigms. Moreover we have upgraded the workflow engine of DIET to take advantage of cloud resources. DIET Cloud will be able to provide a large scale distributed and secured platform that spans on a pool of federated resources that range from dedicated HPC clusters and grid to public and private clouds.

In the context of the Seed4C project, we have studied how secured our platform, authenticated and secured interactions between the different parts of our middleware and between our middleware and its users. We have worked to show how to securely use public cloud storage without taking the risk of losing confidentiality of data stored on them.

5.4. Pilgrim

Participants: Eddy Caron, Matthieu Imbert [correspondant].

Pilgrim (http://pilgrim.gforge.inria.fr) is an open metrology and prediction performance framework whose goal is to provide easy and powerful tools for instrumenting computer platforms and predicting their behavior. Those tools are aimed at being used not only by humans but also by programs, in particular by resource managers and schedulers. Pilgrim is designed to be a loosely coupled integration of various custom-developed or off-the-shelf tools.

5.5. SimGrid

Participants: Georges Markomanolis, Jonathan Rouzaud-Cornabas, Frédéric Suter [correspondant].

SimGrid is a toolkit for the simulation of distributed applications in heterogeneous distributed environments. The specific goal of the project is to facilitate research in the area of parallel and distributed large scale systems, such as Grids, P2P systems and clouds. Its use cases encompass heuristic evaluation, application prototyping or even real application development and tuning. SimGrid has an active user community of more than one hundred members, and is available under GPLv3 from http://simgrid.gforge.inria.fr/.

5.6. HLCMi & L2C

Participants: Zhengxiong Hou, Cristian Klein, Vincent Lanore, Christian Pérez [correspondant], Vincent Pichon.

HLCMI (http://hlcm.gforge.inria.fr) is an implementation of the HLCM component model. HLCM is a generic extensible component model with respect to component implementations and interaction concerns. Moreover, HLCM is abstract; it is its specialization—such as HLCM/L^{2}C—that defines the primitive elements of the model, such as the primitive components and the primitive interactions.

HLCMI is making use of Model-driven Engineering (MDE) methodology to generate a concrete assembly from an high level description. It is based on the Eclipse Modeling Framework (EMF). HLCMI contains 700 Emfatic lines to describe its models and 7000 JAVA lines for utility and model transformation purposes. HLCMI is a general framework that supports several HLCM specializations: HLCM/CCM, HLCM/JAVA, HLCM/L2C and HLCM/Charm++ (known as Gluon++).

L^{2}C (http://hlcm.gforge.inria.fr) is a *Low Level Component* model implementation targeting at use-cases where overhead matters such as High-Performance Computing. L^{2}C does not offer network transparency neither language transparency. Instead, L^{2}C lets the user choose between various kinds of interactions between components, some with ultra low overhead and others that support network transport. L^{2}C is extensible as additional interaction kinds can be added quite easily. L^{2}C currently supports C++, MPI and CORBA interactions. FORTRAN will be added in 2013.

L^{2}C and Gluon++ are implemented in the LLCMc++ framework (http://hlcm.gforge.inria.fr). It is distributed under a LGPL licence and represents 6400 lines of C++.

CEPAGE Project-Team

5. Software

5.1. SimGrid

Participants: Przemyslaw Uznanski, Lionel Eyraud-Dubois [correspondant].

SimGrid (http://simgrid.gforge.inria.fr/) SimGrid is a toolkit that provides core functionalities for the simulation of distributed applications in heterogeneous distributed environments. The specific goal of the project is to facilitate research in the area of parallel and distributed large scale systems, such as Grids, P2P systems and clouds. Its use cases encompass heuristic evaluation, application prototyping or even real application development and tuning. It is based on experimentally validated models, and features very high scalability, which allows to perform very large scale simulations. It is used by over a hundred academic users all over the world, and has been used in about one hundred scientific articles.

CEPAGE has contributed to this software by participating in the management of the project and in many design decisions. We also implemented simpler models, based on our works in this area, allowing a better scalability while keeping a reasonable precision.

Software assessment : A-4, SO-4, SM-4, EM-3, SDL-5.
Contribution : DA-2, CD-2, MS-2, TPM-3.

5.2. Hubble

Participants: Ludovic Courtès, Nicolas Bonichon [correspondant].

Hubble is implemented in Scheme, using GNU Guile version 2. Details of the simulation, such as keeping track of processor occupation and network usage, are taken care of by SimGrid, a toolkit for the simulation of distributed applications in heterogeneous distributed environments.

The input to Hubble is an XML description of the DAG of build tasks. For each task, a build duration and the size in bytes of the build output are specified. For our evaluation purposes, we collected this data on a production system, the http://hydra.nixos.org/ build farm hosted at the Technical University of Delft. The DAG itself is the snapshot of the Nix Package Collection (Nixpkgs) corresponding to this data. Hubble has its own in-memory representation of the DAG in the form of a purely functional data structure.

The Nixpkgs DAG contains fixed-output nodes, i.e., nodes whose output is known in advance and does not require any computation. These nodes are typically downloads of source code from external web sites. The raw data collected on http://hydra.nixos.org/ specifies a non-zero duration for these nodes, which represents the time it took to perform the download. This duration info is irrelevant in our context, since they don't require any computation, and Hubble views these nodes as instantaneous.

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

Software assessment: A-3, SO-3, SM-2, EM-1, SDL-2.
Contribution: DA-4, CD-4, MS-4, TPM-4.

5.3. Gengraph

Participant: Cyril Gavoille [correspondant].

This is a command-line tool for generating graphs. There are several output formats, includes the dot format from GraphViz. It generates also .pdf files for visualization. Several graph algorithms have been implemented (diameter, connectivity, treewidth, etc.) which can be tested on the graphs. The software has been originally designed for teaching purpose so that students can test their project algorithms on many non trivial families like random geometric graphs, graphs of given density, given treewidth. It is also used for research purpose, in particular the exhaustive search results in the Emilie Diot's thesis are based on `gengraph`. The program can filter a list of graphs based to many criteria, as for instance it can extract all graphs of a given list that are 2-connected, of diameter at least four, and that exclude some minor (or some induced subgraph).

Currently, more than 100 parametrized graph families are implemented, supporting simple operators like complementation, random edge/vertex removal, and others. The source has more than 10,000 lines including a command-line documentation of 2,000 lines. The single source file is available at http://dept-info.labri.fr/~gavoille/gengraph.c

Software assessment: A-3, SO-3, SM-2, EM-2, SDL-2.
Contribution: DA-4, CD-4, MS-4, TPM-4.

5.4. Bedibe

Participants: Lionel Eyraud-Dubois [correspondant], Przemyslaw Uznanski.

Bedibe (Benchmarking Distributed Bandwidth Estimation) is a software to compare different models for bandwidth estimation on the Internet, and their associated instantiation algorithms. The goal is to ease the development of new models and algorithms, and the comparison with existing solutions. The development of this software is just starting.

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

Software assessment : A-1-up2, SO-3, SM-1-up2, EM-2, SDL-1-up2.

5.5. MineWithRounds

Participants: Sofian Maabout [correspondant], Nicolas Hanusse.

The software implements a parallel algorithm aiming at computing *Borders* that's sets of maximal/minimal subsets of objects satisfying some anti-monotone condition. It is implemented in `C++` together with the `openMP` library to exploit multi-core machines. In its current status, it outperforms state of the art implementations addressing the Maximal Frequent Itemsets problem.

Software assessment: A-2, SO-4, SM-2, EM-2, SDL-2.
Contribution: DA-4, CD-4, MS-4, TPM-4.

GRAND-LARGE Project-Team

4. Software

4.1. APMC-CA

Participants: Sylvain Peyronnet [correspondant], Joel Falcou, Pierre Esterie, Khaled Hamidouche, Alexandre Borghi.

The APMC model checker implements the state-of-the-art approximate probabilistic model checking methods. Last year we develop a version of the tool dedicated to the CELL architecture. Clearly, it was very pedagogic, but the conclusion is that the CELL is not adapted to sampling based verification methods.

This year we develop, thanks to the BSP++ framework, a version compatible with SPM/multicores machines, clusters and hybrid architectures. This version outperforms all previous ones, thus showing the interest of both these new architectures and of the BSP++ framework.

4.2. YML

Participants: Serge Petiton [correspondant], Nahid Emad, Maxime Hugues.

Scientific end-users face difficulties to program P2P large scale applications using low level languages and middleware. We provide a high level language and a set of tools designed to develop and execute large coarse grain applications on peer-to-peer systems. Thus, we introduced, developed and experimented the YML for parallel programming on P2P architectures. This work was done in collaboration with the PRiSM laboratory (team of Nahid Emad).

The main contribution of YML is its high level language for scientific end-users to develop parallel programs for P2P platforms. This language integrates two different aspects. The first aspect is a component description language. The second aspect allows to link components together. A coordination language called YvetteML can express graphs of components which represent applications for peer-to-peer systems.

Moreover, we designed a framework to take advantage of the YML language. It is based on two component catalogues and an YML engine. The first one concerns end-user's components and the second one is related to middleware criteria. This separation enhances portability of applications and permits real time optimizations. Currently we provide support for the XtremWeb Peer-to-Peer middleware and the OmniRPC grid system. The support for Condor is currently under development and a beta-release will be delivered soon (in this release, we plan to propagate semantic data from the end-users to the middleware). The next development of YML concerns the implementation of a multi-backend scheduler. Therefore, YML will be able to schedule at runtime computing tasks to any global computing platform using any of the targeted middleware.

We experimented YML with basic linear algebra methods on a XtremWeb P2P platform deployed between France and Japan. Recently, we have implemented complex iterative restarted Krylov methods, such as Lanczos-Bisection, GMRES and MERAM methods, using YML with the OmniRPC back-end. The experiments are performed either on the Grid5000 testbed of on a Network of Workstations deployed between Lille, Versailles and Tsukuba in Japan. Demos was proposed on these testbeds from conferences in USA. We recently finished evaluations of the overhead generated using YML, without smart schedulers and with extrapolations due to the lack of smart scheduling strategies inside targeted middleware.

In the context of the FP3C project funded by ANR-JST, we have recently extended YML to support a directive distributed parallel language, XcalableMP http://www.xcalablemp.org/. This extension is based on the support of the XcalableMP language inside YML components. This allows to develop parallel programs with two programming paradigm and thus two parallelism levels. This work is a part of the project that targets post-Petascale supercomputer that would be composed of heterogeneous and massively parallel hardware.

The software is available at http://yml.prism.uvsq.fr/

4.3. The Scientific Programming InterNet (SPIN)

Participant: Serge Petiton [correspondant].

SPIN (Scientific Programming on the InterNet), is a scalable, integrated and interactive set of tools for scientific computations on distributed and heterogeneous environments. These tools create a collaborative environment allowing the access to remote resources.

The goal of SPIN is to provide the following advantages: Platform independence, Flexible parameterization, Incremental capacity growth, Portability and interoperability, and Web integration. The need to develop a tool such as SPIN was recognized by the GRID community of the researchers in scientific domains, such as linear algebra. Since the P2P arrives as a new programming paradigm, the end-users need to have such tools. It becomes a real need for the scientific community to make possible the development of scientific applications assembling basic components hiding the architecture and the middleware. Another use of SPIN consists in allowing to build an application from predefined components ("building blocks") existing in the system or developed by the developer. The SPIN users community can collaborate in order to make more and more predefined components available to be shared via the Internet in order to develop new more specialized components or new applications combining existing and new components thanks to the SPIN user interface.

SPIN was launched at ASCI CNRS lab in 1998 and is now developed in collaboration with the University of Versailles, PRiSM lab. SPIN is currently under adaptation to incorporate YML, cf. above. Nevertheless, we study another solution based on the Linear Algebra KErnel (LAKE), developed by the Nahid Emad team at the University of Versailles, which would be an alternative to SPIN as a component oriented integration with YML.

4.4. V-DS

Participant: Franck Cappello [correspondant].

This project started officially in September 2004, under the name V-Grid. V-DS stands for Virtualization environment for large-scale Distributed Systems. It is a virtualization software for large scale distributed system emulation. This software allows folding a distributed systems 100 or 1000 times larger than the experimental testbed. V-DS virtualizes distributed systems nodes on PC clusters, providing every virtual node its proper and confined operating system and execution environment. Thus compared to large scale distributed system simulators or emulators (like MicroGrid), V-DS virtualizes and schedules a full software environment for every distributed system node. V-DS research concerns emulation realism and performance.

A first work concerns the definition and implementation of metrics and methodologies to compare the merits of distributed system virtualization tools. Since there is no previous work in this domain, it is important to define what and how to measure in order to qualify a virtualization system relatively to realism and performance. We defined a set of metrics and methodologies in order to evaluate and compared virtualization tools for sequential system. For example a key parameter for the realism is the event timing: in the emulated environment, events should occur with a time consistent with a real environment. An example of key parameter for the performance is the linearity. The performance degradation for every virtual machine should evolve linearly with the increase of the number of virtual machines. We conducted a large set of experiments, comparing several virtualization tools including Vserver, VMware, User Mode Linux, Xen, etc. The result demonstrates that none of them provides both enough isolation and performance. As a consequence, we are currently studying approaches to cope with these limits.

We have made a virtual platform on the GDX cluster with the Vserver virtualization tool. On this platform, we have launched more than 20K virtual machines (VM) with a folding of 100 (100 VM on each physical machine). However, some recent experiments have shown that a too high folding factor may cause a too long execution time because of some problems like swapping. Currently, we are conducting experiments on another platform based on the virtualization tool named Xen which has been strongly improved since 2 years. We expect to get better result with Xen than with Vserver. Recently, we have been using the V-DS version based on Xen to evaluate at large scales three P2P middleware [83].

This software is available at http://v-ds.lri.fr/

4.5. PVC: Private Virtual Cluster

Participant: Franck Cappello [correspondant].

Current complexity of Grid technologies, the lack of security of Peer-to-Peer systems and the rigidity of VPN technologies make sharing resources belonging to different institutions still technically difficult.

We propose a new approach called "Instant Grid" (IG), which combines various Grid, P2P and VPN approaches, allowing simple deployment of applications over different administration domains. Three main requirements should be fulfilled to make Instant Grids realistic: simple networking configuration (Firewall and NAT), no degradation of resource security, no need to re-implement existing distributed applications.

Private Virtual Cluster, is a low-level middle-ware that meets Instant Grid requirements. PVC turns dynamically a set of resources belonging to different administration domains into a virtual cluster where existing cluster runtime environments and applications can be run. The major objective of PVC is to establish direct connections between distributed peers. To connect firewall protected nodes in the current implementation, we have integrated three techniques: UPnP, TCP/UDP Hole Punching and a novel technique Traversing-TCP.

One of the major application of PVC is the third generation desktop Grid middleware. Unlike BOINC and XtremWeb (which belong to the second generation of desktop Grid middleware), PVC allows the users to build their Desktop Grid environment and run their favorite batch scheduler, distributed file system, resource monitoring and parallel programming library and runtime software. PVC ensures the connectivity layer and provide a virtual IP network where the user can install and run existing cluster software.

By offering only the connectivity layer, PVC allows to deploy P2P systems with specific applications, like file sharing, distributed computing, distributed storage and archive, video broadcasting, etc.

4.6. OpenWP

Participant: Franck Cappello [correspondant].

Distributed applications can be programmed on the Grid using workflow languages, object oriented approaches (Proactive, IBIS, etc), RPC programming environments (Grid-RPC, DIET), component based environments (generally based on Corba) and parallel programming libraries like MPI.

For high performance computing applications, most of the existing codes are programmed in C, Fortran and Java. These codes have 100,000 to millions of lines. Programmers are not inclined to rewrite then in a "non standard" programming language, like UPC, CoArray Fortran or Global Array. Thus environments like MPI and OpenMPI remain popular even if they require hybrid approaches for programming hierarchical computing infrastructures like cluster of multi-processors equipped with multi-core processors.

Programming applications on the Grid add a novel level in the hierarchy by clustering the cluster of multi-processors. The programmer will face strong difficulties in adapting or programming a new application for these runtime infrastructures featuring a deep hierarchy. Directive based parallel and distributed computing is appealing to reduce the programming difficulty by allowing incremental parallelization and distribution. The programmer add directives on a sequential or parallel code and may check for every inserted directive its correction and performance improvement.

We believe that directive based parallel and distributed computing may play a significant role in the next years for programming High performance parallel computers and Grids. We have started the development of OpenWP. OpenWP is a directive based programming environment and runtime allowing expressing workflows to be executed on Grids. OpenWP is compliant with OpenMP and can be used in conjunction with OpenMP or hybrid parallel programs using MPI + OpenMP.

The OpenWP environment consists in a source to source compiler and a runtime. The OpenWP parser, interprets the user directives and extracts functional blocks from the code. These blocks are inserted in a library distributed on all computing nodes. In the original program, the functional blocks are replaced by RPC calls and calls to synchronization. During the execution, the main program launches non blocking RPC calls to functions on remote nodes and synchronize the execution of remote functions based on the synchronization directives inserted by the programmer in the main code. Compared to OpenMP, OpenWP does not consider a shared memory programming approach. Instead, the source to source compiler insert data movements calls in the main code. Since the data set can be large in Grid application, the OpenWP runtime organize the storage of data sets in a distributed way. Moreover, the parameters and results of RPC calls are passed by reference, using a DHT. Thus, during the execution, parameter and result references are stored in the DHT along with the current position of the datasets. When a remote function is called, the DHT is consulted to obtain the position of the parameter data sets in the system. When a remote function terminates its execution, it stores the result data sets and store a reference to the data set in the DHT.

We are evaluating OpenWP from an industrial application (Amibe), used by the European aerospace company EADS. Amibe is the mesher module of jCAE [1]. Amibe generates a mesh from a CAD geometry in three steps. It first creates edges between every patch of the CAD (mesh in one dimension), then generates a surface mesh for every unfolded patch (mesh in two dimensions) and finally adds the third dimension to the mesh by projecting the 2D mesh into the original CAD surfaces. The first and third operation cannot be distributed. However the second step can easily be distributed following a master/worker approach, transferring the mesh1d results to every computing node and launching the distributed execution of the patches.

4.7. Parallel solvers for solving linear systems of equations

Participant: Laura Grigori.

In the last several years, there has been significant research effort in the development of fully parallel direct solvers for computing the solution of large unsymmetric sparse linear systems of equations. In this context, we have designed and implemented a parallel symbolic factorization algorithm, which is suitable for general sparse unsymmetric matrices. The symbolic factorization is one of the steps that is sequential and represents a memory bottleneck. The code is intended to be used with very large matrices when because of the memory usage, the sequential algorithm is not suitable. This code is available in the SuperLU_DIST, a widely used software, developed at UC Berkeley and LBNL by Professor James W. Demmel and Dr. Xiaoye S. Li. The algorithm is presented in [72]. The SuperLU_DIST is available at http://crd.lbl.gov/~xiaoye/SuperLU/ .

4.8. OpenScop

Participant: Cédric Bastoul.

OpenScop is an open specification which defines a file format and a set of data structures to represent a *static control part* (SCoP for short), i.e., a program part that can be represented in the *polyhedral model*, an algebraic representation of programs used for automatic parallelization and optimization (used, e.g., in GNU GCC, LLVM, IBM XL or Reservoir Labs R-Stream compilers). The goal of OpenScop is to provide a common interface to various polyhedral compilation tools in order to simplify their interaction.

OpenScop provides a single format for tools that may have different purposes (e.g., as different as code generation and data dependence analysis). We could observe that most available polyhedral compilation tools during the last decade were manipulating the same kind of data (polyhedra, affine functions...) and were actually sharing a part of their input (e.g., iteration domains and context concepts are nearly everywhere). We could also observe that those tools may rely on different internal representations, mostly based on one of the major polyhedral libraries (e.g., Polylib, PPL or isl), and this representation may change over time (e.g., when switching to a more convenient polyhedral library). OpenScop aims at providing a stable, unified format that offers a durable guarantee that a tool can use an output or provide an input to another tool without breaking a compilation chain because of some internal changes in one element of this chain. The other promise of

[1] project page: http://jcae.sourceforge.net

OpenScop is the ability to assemble or replace the basic blocks of a polyhedral compilation framework at no, or at least low engineering cost. The OpenScop Library (licensed under the 3-clause BSD license) has been developped as an example, yet powerful, implementation of the OpenScop specification.

4.9. Clay

Participant: Cédric Bastoul.

Clay is a free software and library devoted to semi-automatic optimization using the polyhedral model. It can input a high-level program or its polyhedral representation and transform it according to a transformation script. Classic loop transformations primitives are provided. Clay is able to check for the legality of the complete sequence of transformation and to suggest corrections to the user if the original semantics is not preserved (experimental at this document redaction time). Main authors include Joël Poudroux and Cédric Bastoul.

4.10. CALU for multicore architectures

Participant: Laura GRIGORI [correspondant].

The communication avoiding algorithms are implemented in the form of a portable library. In its current form, this library is designed for multicore architectures and uses a hybrid scheduling technique that exploits well the data locality and can adapt to dynamic changes in the machine. The library will be publicly available since February 2012.

See also the web page http://www-rocq.inria.fr/who/Laura.Grigori/COALA2010/coala.html.

- Version: 1.0

4.11. MIDAPACK for CMB data analysis

Participants: Laura GRIGORI [correspondant], Mikolaj SZYDLARSKI [correspondant].

Midapack is a library aiming at the crucial stages down the CMB data analysis pipeline. In its current form, the library provides tools for computing spherical harmonic transforms on heterogeneous architectures, and algorithms for finding the solution to a generalized least squares problem. The algorithms are described in [36] and [44]. See also the web page http://pages.saclay.inria.fr/laura.grigori/soft.html.

- Version: 1.0

4.12. Fast linear system solvers in public domain libraries

Participant: Marc Baboulin [correspondant].

Hybrid multicore+GPU architectures are becoming commonly used systems in high performance computing simulations. In this research, we develop linear algebra solvers where we split the computation over multicore and graphics processors, and use particular techniques to reduce the amount of pivoting and communication between the hybrid components. This results in efficient algorithms that take advantage of each computational unit [14]. Our research in randomized algorithms yields to several contributions to propose public domain libraries PLASMA and MAGMA in the area of fast linear system solvers for general and symmetric indefinite systems. These solvers minimize communication by removing the overhead due to pivoting in LU and $LDLT$ factorization. Different approaches to reduce communication are compared in [26].

See also the web page http://icl.cs.utk.edu/magma/.

4.13. cTuning: Repository and Tools for Collective Characterization and Optimization of Computing Systems

Participant: Grigori Fursin [correspondant].

Designing, porting and optimizing applications for rapidly evolving computing systems is often complex, ad-hoc, repetitive, costly and error prone process due to an enormous number of available design and optimization choices combined with the complex interactions between all components. We attempt to solve this fundamental problem based on collective participation of users combined with empirical tuning and machine learning.

We developed cTuning framework that allows to continuously collect various knowledge about application characterization and optimization in the public repository at cTuning.org. With continuously increasing and systematized knowledge about behavior of computer systems, users should be able to obtain scientifically motivated advices about anomalies in the behavior of their applications and possible solutions to effectively balance performance and power consumption or other important characteristics.

Currently, we use cTuning repository to analyze and learn profitable optimizations for various programs, datasets and architectures using machine learning enabled compiler (MILEPOST GCC). Using collected knowledge, we can quickly suggest better optimizations for a previously unseen programs based on their semantic or dynamic features [6].

We believe that such approach will be vital for developing efficient Exascale computing systems. We are currently developing the new extensible cTuning2 framework for automatic performance and power tuning of HPC applications.

For more information, see the web page http://cTuning.org.

HIEPACS Project-Team

5. Software

5.1. Introduction

We describe in this section the software that we are developing. The first two (`MaPHyS` and `ScalFMM`) will be the main milestones of our project. The other software developments will be conducted in collaboration with academic partners or in collaboration with some industrial partners in the context of their private R&D or production activities. For all these software developments, we will use first the various (very) large parallel platforms available through CERFACS and GENCI in France (CCRT, CINES and IDRIS Computational Centers), and next the high-end parallel platforms that will be available via European and US initiatives or projects such that PRACE.

5.2. MaPHyS

`MaPHyS` (Massivelly Parallel Hybrid Solver) is a software package whose proptotype was initially developed in the framework of the PhD thesis of Azzam Haidar (CERFACS) and futher consolidated thanks to the ANR-CIS Solstice funding. This parallel linear solver couples direct and iterative approaches. The underlying idea is to apply to general unstructured linear systems domain decomposition ideas developed for the solution of linear systems arising from PDEs. The interface problem, associated with the so called Schur complement system, is solved using a block preconditioner with overlap between the blocks that is referred to as Algebraic Additive Schwarz.

The `MaPHyS` package is very much a first outcome of the research activity described in Section 3.3 . Finally, `MaPHyS` is a preconditioner that can be used to speed-up the convergence of any Krylov subspace method. We forsee to either embed in `MaPHyS` some Krylov solvers or to release them as standalone packages, in particular for the block variants that will be some outcome of the studies discussed in Section 3.3 .

5.3. EPSN

EPSN (Environment for Computational Steering) is a software environment for the steering of legacy parallel-distributed simulations with simple GUI or more complex (possibly parallel) visualization programs (see Figure 1). In order to make a legacy simulation steerable, the user annotates the sourcecode with the EPSN API. These annotations provide the EPSN environment with two kinds of information: the description of the program structure according to a Hierarchical Task Model (HTM) and the description of the distributed data that will be remotely accessible. EPSN provides a distributed data model, that handles common scientific objects such as parameters, structured grids, particles/atoms and unstructured meshes. It is then possible to dynamically connect EPSN with a client program, that provides a GUI with some visualization & interaction features, as for instance SIMONE (SImulation MONitoring for Epsn). Once a client is connected, it interacts with the simulation via EPSN API. It is possible : 1) to control the execution flow of the remote simulation; 2) to access/modify its data onthefly; and 3) finally to invoke advanced user-defined routines in the simulation. The current version of EPSN is fully based on CORBA for communication on heterogeneous system and VTK/Paraview for visualization. A new release of EPSN, that will be fully based on MPI to handle efficient communication, is currently under development. A prototype is already working.

EPSN has been supported by the ACI-GRID program (grant number PPL02-03), the ARC RedGRID, the ANR MASSIM (grant number ANR-05-MMSA-0008-03) and the ANR CIS NOSSI (2007). More informations are available on our web site: http://www.labri.fr/projet/epsn. This software is publicy available at Inria Gforge (http://epsn.gforge.inria.fr).

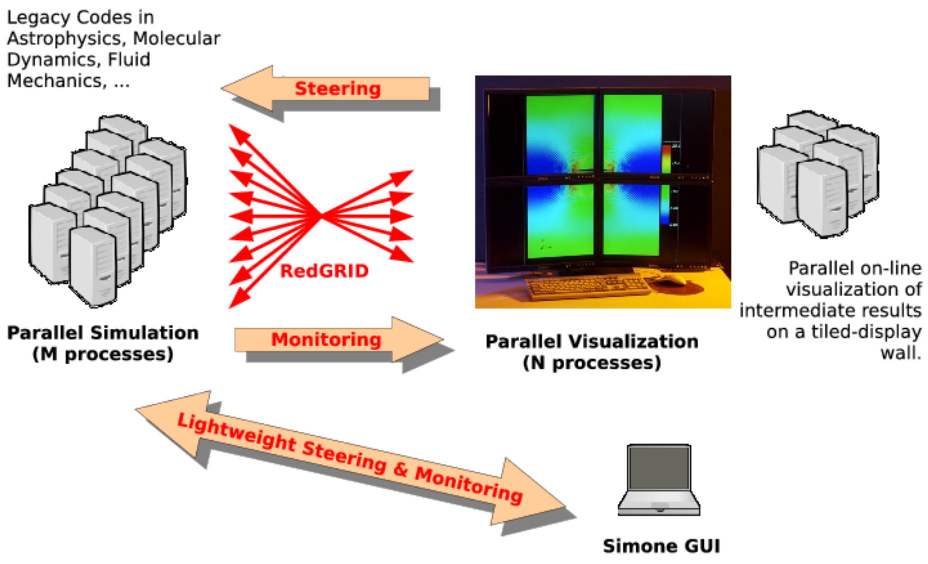

Legacy Codes in Astrophysics, Molecular Dynamics, Fluid Mechanics, ...
Steering
RedGRID
Parallel Simulation (M processes)
Monitoring
Parallel Visualization (N processes)
Parallel on-line visualization of intermediate results on a tiled-display wall.
Lightweight Steering & Monitoring
Simone GUI

Figure 1. EPSN: software environment for $M \times N$ computational steering.

5.4. MPICPL

MPICPL (MPI CouPLing) is a software library dedicated to the coupling of parallel legacy codes, that are based on the well-known MPI standard. It proposes a lightweight and comprehensive programing interface that simplifies the coupling of several MPI codes (2, 3 or more). MPICPL facilitates the deployment of these codes thanks to the *mpicplrun* tool and it interconnects them automatically through standard MPI inter-communicators. Moreover, it generates the universe communicator, that merges the world communicators of all coupled-codes. The coupling infrastructure is described by a simple XML file, that is just loaded by the *mpicplrun* tool. Future releases will incorporate new features for checkpoint/restart and dynamic parallel code connection.

MPICPL was developed by the Inria HiePACS project-team for the purpose of the ANR CIS NOSSI. It uses advanced features of MPI2 standard. The framework is publicy available at Inria Gforge: http://mpicpl.gforge.inria.fr.

5.5. MONIQA

MONIQA (MONitoring graphic user Interface for QM/MM Applications) is a GUI specially designed for the monitoring & steering of the QM/MM application in the ANR CIS NOSSI project. It is based on Tulip, a graph visualization software http://tulip.labri.fr), used to display atoms and molecules. It proposes two working modes : offline or online. The offline mode is mainly used to load input files of DL_POLY & Siesta, and to prepare the quantum region for the QM/MM coupling. In online mode, the end-user can monitor & interact with the running QM/MM application thanks to EPSN. It is thus possible to visualize molecular and physical data (distances, angles, charges, energies), and to change simulation parameters on-the-fly, such as the target temperature of the system, thermo or barostat parameters, verbosity of output, ... MONIQA is based on QT4. It was developed specifically for the ANR NOSSI project and is available (restricted access) at Inria Gforge: http://nossi.gforge.inria.fr.

5.6. ScalFMM

`ScalFMM` intends to offer all the functionalities needed to perform large parallel simulations while enabling an easy customization of the simulation components: kernels, particles and cells. It works in parallel in a shared/distributed memory model using OpenMP and MPI. The software architecture has been designed with two major objectives: being easy to maintain and easy to understand. The code is extremely documented and the naming convention fully respected. Driven by its user-oriented philosophy, `ScalFMM` is using CMAKE as a compiler/installer tool. Even if `ScalFMM` is written in C++ it will support a C and fortran API soon.

`ScalFMM` (Parallel Fast Multipole Library for Large Scale Simulations) is a software library to simulate N-body interactions using the Fast Multipole Method.

The library offers two methods to compute interactions between bodies when the potential decays like $1/r$. The first method is the classical FMM based on spherical harmonic expansions and the second is the Black-Box method which is an independent kernel formulation (introduced by E. Darve @ Stanford). With this method, we can now easily add new non oscillatory kernels in our library. For the classical method, two approaches are used to decrease the complexity of the operators. We consider either matrix formulation that allows us to use BLAS routines or rotation matrix to speed up the M2L operator.

`ScalFMM` intends to offer all the functionalities needed to perform large parallel simulations while enabling an easy customization of the simulation components: kernels, particles and cells. It works in parallel in a shared/distributed memory model using OpenMP and MPI. The software architecture has been designed with two major objectives: being easy to maintain and easy to understand. There is two main parts: 1) the management of the octree and the parallelization of the method ; 2) the kernels. This new architecture allow us to easily add new FMM algorithm or kernels and new paradigm of parallelization.

The `ScalFMM` package is available at scalfmm.gforge.inria.fr.

5.7. Other software

These software packages are or will be developed in collaboration with some academic partners (LIP6, LaBRI, CPMOH, IPREM, EPFL) or in collaboration with industrial partners (CEA, TOTAL, EDF) in the context of their private R&D or production activities.

- For the materials physics applications, a lot of development will be done in the context of ANR projects (NOSSI and OPTIDIS, see Section 4.2) in collaboration with LaBRI, CPMOH, IPREM, EPFL and with CEA Saclay and Bruyère-le-Châtel.

KERDATA Project-Team

5. Software

5.1. BlobSeer

Participants: Viet-Trung Tran, Zhe Li, Alexandru Costan, Gabriel Antoniu, Luc Bougé.

Contact: Gabriel Antoniu.

Presentation: BlobSeer is the core software platform for most current projects of the KerData team. It is a data storage service specifically designed to deal with the requirements of large-scale data-intensive distributed applications that abstract data as huge sequences of bytes, called BLOBs (Binary Large OBjects). It provides a versatile versioning interface for manipulating BLOBs that enables reading, writing and appending to them.

BlobSeer offers both scalability and performance with respect to a series of issues typically associated with the data-intensive context: *scalable aggregation of storage space* from the participating nodes with minimal overhead, ability to store *huge data objects*, *efficient fine-grain access* to data subsets, *high throughput in spite of heavy access concurrency*, as well as *fault-tolerance*.

Users: Work is currently in progress in several formalized projects (see previous section) to integrate and leverage BlobSeer as a data storage back-end in the reference cloud environments: a) Microsoft Azure; b) the Nimbus cloud toolkit developed at Argonne National Lab (USA); and c) in the OpenNebula IaaS cloud environment developed at UCM (Madrid).

URL: http://blobseer.gforge.inria.fr/

License: GNU Lesser General Public License (LGPL) version 3.

Status: This software is available on Inria's forge. Version 1.0 (released late 2010) registered with APP: IDDN.FR.001.310009.000.S.P.000.10700.

A new *Technology Research Action* (ADT, *Action de recherche technologique*) has been launched in Septembre 2012 for one year, with a possible 1-year renewal, to robustify the BlobSeer software and and make it a safeky distributable product. This project is funded by Inria *Technological Development Office* (D2T, *Direction du Développement Technologique*). Zhe Li has been hired as a senior (PhD) engineer for this task.

5.2. Damaris

Participants: Matthieu Dorier, Gabriel Antoniu.

Contact: Gabriel Antoniu.

Presentation: Damaris is a middleware for multicore SMP nodes enabling them to efficiently handle data transfers for storage and visualization. The key idea is to dedicate one or a few cores of each SMP node to the application I/O. It is developed within the framework of a collaboration between KerData and the Joint Laboratory for Petascale Computing (JLPC). The current version enables efficient asynchronous I/O, hiding all I/O related overheads such as data compression and post-processing. On-going work is targeting fast direct access to the data from running simulations, and efficient I/O scheduling.

Users: Damaris has been preliminarily evaluated at NCSA (Urbana-Champaign) with the CM1 tornado simulation code. CM1 is one of the target applications of the Blue Waters supercomputer developed by at NCSA/UIUC (USA), in the framework of the Inria-UIUC-ANL Joint Lab (JLPC). Damaris now has external users, including (to our knowledge) visualization specialists from NCSA and researchers from the France/Brazil Associated research team on Parallel Computing (joint team between Inria/LIG Grenoble and the UFRGS in Brazil). Damaris has been successfully integrated into three large-scale simulations (CM1, OLAM, Nek5000). Works are in progress to evaluate it in the context of several other simulations including HACC (cosmology code) and GTC (fusion).

URL: http://damaris.gforge.inria.fr/

License: GNU Lesser General Public License (LGPL) version 3.

Status: This software is available on Inria's forge. Registration with APP is in progress.

5.3. Derived software

Derived from BlobSeer, two additional platforms are currently being developed within KerData: 1) Pyramid, a software service for array-oriented active storage developed within the framework of the PhD thesis of Viet-Trung Tran; and 2) BlobSeer-WAN, a data management service specifically optimized for geographically distributed environments. It is also developed within the framework of the PhD thesis of Viet-Trung Tran in relation to the FP3C project. These platforms have not been publicly released yet.

MESCAL Project-Team

5. Software

5.1. Tools for cluster management and software development

Participant: Olivier Richard [correspondant].

The KA-Tools is a software suite developped by MESCAL for exploitation of clusters and grids. It uses a parallelization technique based on spanning trees with a recursive starting of programs on nodes. Industrial collaborations were carried out with Mandrake, BULL, HP and Microsoft.

KA-DEPLOY is an environment deployment toolkit that provides automated software installation and reconfiguration mechanisms for large clusters and light grids. The main contribution of KA-DEPLOY 2 toolkit is the introduction of a simple idea, aiming to be a new trend in cluster and grid exploitation: letting users concurrently deploy computing environments tailored exactly to their experimental needs on different sets of nodes. To reach this goal KA-DEPLOY must cooperate with batch schedulers, like OAR, and use a parallel launcher like TAKTUK (see below).

TAKTUK is a tool to launch or deploy efficiently parallel applications on large clusters, and simple grids. Efficiency is obtained thanks to the overlap of all independent steps of the deployment. We have shown that this problem is equivalent to the well known problem of the single message broadcast. The performance gap between the cost of a network communication and of a remote execution call enables us to use a work stealing algorithm to realize a near-optimal schedule of remote execution calls. Currently, a complete rewriting based on a high level language (precisely Perl script language) is under progress. The aim is to provide a light and robust implementation. This development is lead by the MOAIS project-team.

5.2. OAR: Batch scheduler for clusters and grids

Participant: Olivier Richard [correspondant].

The OAR project focuses on robust and highly scalable batch scheduling for clusters and grids. Its main objectives are the validation of grid administration tools such as TAKTUK, the development of new paradigms for grid scheduling and the experimentation of various scheduling algorithms and policies.

The grid development of OAR has already started with the integration of best effort jobs whose purpose is to take advantage of idle times of the resources. Managing such jobs requires a support of the whole system from the highest level (the scheduler has to know which tasks can be canceled) down to the lowest level (the execution layer has to be able to cancel awkward jobs). The OAR architecture is perfectly suited to such developments thanks to its highly modular architecture. Moreover, this development is used for the CiGri grid middleware project.

The OAR system can also be viewed as a platform for the experimentation of new scheduling algorithms. Current developments focus on the integration of theoretical batch scheduling results into the system so that they can be validated experimentally.

See also the web page http://oar.imag.fr.

5.3. CiGri: Computing ressource Reaper

Participant: Olivier Richard [correspondant].

CiGri is a middleware which gather the unused computing resource from intranet infrastructure and to make it available for large set of tasks. It manages the execution of large sets of parametric tasks on lightweight grid by submitting individual jobs to each batch scheduler. It's associated to the OAR resource management system (batch scheduler). Users can easily monitor and control their set of jobs through a web portal. System provides mechanisms to identify job error causes, to isolate faulty components and to resubmit job in a safer context. See also the web page http://cigri.imag.fr/

5.4. FTA: Failure Trace Archive

Participant: Derrick Kondo [correspondant].

The Failure Trace Archive is available at http://fta.inria.fr.

With the increasing functionality, scale, and complexity of distributed systems, resource failures are inevitable. While numerous models and algorithms for dealing with failures exist, the lack of public trace data sets and tools has prevented meaningful comparisons. To facilitate the design, validation, and comparison of fault-tolerant models and algorithms, we led the creation of the Failure Trace Archive (FTA), an on-line public repository of availability traces taken from diverse parallel and distributed systems.

While several archives exist, the FTA differs in several respects. First, it defines a standard format that facilitates the use and comparison of traces. Second, the archive contains traces in that format for over 20 diverse systems over a time span of 10 years. Third, it provides a public toolbox for failure trace interpretation, analysis, and modeling. The FTA was released in November 2009. It has received over 11,000 hits since then. The FTA has had national and international impact. Several published works have already cited and benefited from the traces and tools of the FTA. Simulation toolkits for distributed systems, such as SimGrid (CNRS, France) and GridSim (University of Melbourne, Australia), have incorporated the traces to allow for simulations with failures.

5.5. SimGrid: simulation of distributed applications

Participants: Arnaud Legrand [correspondant], Lucas Schnorr, Pierre Navarro, Degomme Augustin, Laurent Bobelin.

SimGrid is a toolkit that provides core functionalities for the simulation of distributed applications in heterogeneous distributed environments. The specific goal of the project is to facilitate research in the area of distributed and parallel application scheduling on distributed computing platforms ranging from simple network of workstations to Computational Grids.

We have released one new major version (3.6) of SimGrid (June 2011) and two minor versions (June and October 2011). These versions include our current work on visualization, analysis of large scale distributed systems, and extremely scalable simulation. See also the web page http://simgrid.gforge.inria.fr/.

5.6. TRIVA: interactive trace visualization

Participants: Lucas Schnorr [correspondant], Arnaud Legrand.

TRIVA is an open-source tool used to analyze traces (in the Pajé format) registered during the execution of parallel applications. The tool serves also as a sandbox for the development of new visualization techniques. Some features include: Temporal integration using dynamic time-intervals; Spatial aggregation through hierarchical traces; Scalable visual analysis with squarified treemaps; A Custom Graph Visualization.

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

5.7. ψ and ψ^2: perfect simulation of Markov Chain stationary distributions

Participant: Jean-Marc Vincent [correspondant].

ψ and ψ^2 are two software tools implementing perfect simulation of Markov Chain stationary distributions using *coupling from the past*. ψ starts from the transition kernel to derive the simulation program while ψ^2 uses a monotone constructive definition of a Markov chain. They are available at http://www-id.imag.fr/Logiciels/psi/.

5.8. GameSeer: simulation of game dynamics

Participant: Panayotis Mertikopoulos [correspondant].

Mathematica toolbox (graphical user interface and functions library) for efficient, robust and modular simulations of game dynamics.

5.9. Kameleon: environment for experiment reproduction

Participants: Olivier Richard [correspondant], Joseph Emeras.

Kameleon is a tool developed to facilitate the building and rebuilding of software environment. It helps experimenter to manage his experiment's software environment which can include the operating system, libraries, runtimes, his applications and data. This tool is an element in the experimental process to obtain repeatable experiments and therefore reproducible results.

MOAIS Project-Team

5. Software

5.1. KAAPI

Participants: Thierry Gautier [correspondant], Vincent Danjean, François Broquedis, Pierre Neyron.

Kaapi (http://kaapi.gforge.inria.fr, coordinator T. Gautier) Kaapi is a middleware for high performance applications running on multi-cores/multi- processors as well as cluster or computational grid. Kaapi provides 1/ a very high level API based on macro data flow language; 2/ several scheduling algorithms for multi-threaded computations as well as for iterative applications for numerical sim- ulation on multi-CPUs / multi-GPUs; 3/ fault-tolerant protocols. Publicly available at http://kaapi.gforge.inria.fr under CeCILL licence. Kaapi has won the 2008 Plugtest organized by Grid@Works. Kaapi provides ABI compliant implementations of Quark (PLASMA, Linear Algebra, Univ. of Tennesse) and libGOMP (GCC runtime for OpenMP). Direct competitors with 1/: Quark, StarSs (UPC, BSC), OpenMP. Direct competitors with 2/: StarSs, StarPU (Inria RUNTIME), Quark, OpenACC runtimes. Direct competitors providing 3/: Charm++, MPI.

- ACM: D.1.3
- License: CeCILL
- OS/Middelware: Unix (Linux, MacOSX, ...)
- Programming language: C/C++, Fortran
- Characterization of Software : A-3 / SO-4 / SM-3 / EM-3 / SDL-4
- Own Contribution: DA-4 / CD-4 / MS-4 / TPM-4

5.2. FlowVR

Participant: Bruno Raffin [correspondant MOAIS].

- Characterization of Software : A-3 / SO-4 / SM-3 / EM-3 / SDL-4
- Own Contribution: DA-4 / CD-3 / MS-3 / TPM-4
- Additional information: FlowVR (http://flowvr.sf.net, coordinator B. Raffin) provides users with the necessary tools to develop and run high performance interactive applications on PC clusters and Grids. The main target applications include virtual reality, scientific visualization and Web3D. FlowVR enforces a modular programming that leverages software engineering issues while enabling high performance executions on distribued and parallel architectures. FlowVR is the reference API for Grimage. See also the web page http://flowvr.sf.net. The FlowVR software suite has 3 main components:
  - FlowVR : The core middleware library. FlowVR relies on the data-flow oriented programming approach that has been successfully used by other scientific visualization tools.
  - FlowVR Render : A parallel rendering library.
  - VTK FlowVR : a VTK / FlowVR / FlowVR Render coupling library.

5.3. TakTuk - Adaptive large scale remote execution deployment

Participants: Guillaume Huard [correspondant], Pierre Neyron.

- Characterization of Software : A-2 / SO-3 / SM-5 / EM-3 / SDL-4
- Own Contribution: DA-4 / CD-4 / MS-4 / TPM-4
- Additional information:
  - web site: http://taktuk.gforge.inria.fr, Coordinator G. Huard
  - Objective of the software: TakTuk is a tool for deploying parallel remote executions of commands to a potentially large set of remote nodes. It spreads itself using an adaptive algorithm and sets up an interconnection network to transport commands and perform I/Os multiplexing/demultiplexing. The TakTuk mechanics dynamically adapt to environment (machine performance and current load, network contention) by using a reactive work-stealing algorithm that mixes local parallelization and work distribution.
  - Users community: TakTuk is a research open source project available in the Debian GNU/Linux distribution (package taktuk) used in lower levels of Grid5000 software architectures (nodes monitoring in OAR, environment diffusion in Kadeploy). The community is small : developers and administrators for large scale distributed platforms, but active.
  - Positioning: main competing tools are pdsh (but uses linear deployment) and gexec (not fault tolerant, requires installation), for more details : B. Claudel, G. Huard and O. Richard. TakTuk, Adaptive Deployment of Remote Executions. In Proceedings of the International Symposium on High Performance Distributed Computing (HPDC), 2009. TakTuk is the only tool to provide to deployed processes a communication layer (just like an MPIrun, but not tied to a specific environment) and synchronization capabilities.

5.4. KRASH - Kernel for Reproduction and Analysis of System Heterogeneity

Participants: Guillaume Huard [correspondant], Swann Perarnau.

- Characterization of Software : A-1 / SO-3 / SM-4 / EM-2 / SDL-3
- Own Contribution: DA-4 / CD-4 / MS-4 / TPM-4
- Additional information:
  - web site: http://krash.ligforge.imag.fr
  - Objective of the software: Krash is a tool to create a synthetic heterogeneity on top of a dedicated system while preserving the OS state and algorithms (no modification). It makes use of the control groups (cgroups) in Linux kernel newer than version 2.6.24 to create a dynamic CPU load enforced no matter how many applications are running in parallel.
  - Users community: Research open source project, small community: developers of parallel applications in heterogeneous contexts.
  - Positioning: Competing tool is Wreakavoc (less scalable, less precise), more details in : Swann Perarnau and Guillaume Huard. Krash: Reproducible cpu load generation on many-core machines. In IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2010.

5.5. Cache Control

Participants: Guillaume Huard [correspondant], Swann Perarnau.

- Characterization of Software : A-1 / SO-3 / SM-3 / EM-2 / SDL-3
- Own Contribution: DA-4 / CD-4 / MS-4 / TPM-4
- Additional information:
  – web site: http://ccontrol.ligforge.imag.fr/
  – Objective of the software: Cache Control is a Linux kernel module enabling user applications to restrict their memory allocations to a subset of the hardware memory cache. This module reserves and exports available physical memory as virtual devices that can be mmap'd to. It gives to calling processes physical memory using only a subset of the cache (similarly to page coloring). It actually creates cache partitions that can be used simultaneously by a process to control how much cache a data structure can use.
  – Users community: Research open source project, small community: developers wanting to measure or tune the cache usage of their applications. Does not apply to recent NUCA caches.
  – Positioning: Competing tool is ULCC which does the same thing at the runtime level, more details in : Swann Perarnau, Marc Tchiboukdjian, and Guillaume Huard. Controlling cache utilization of hpc applications. In International Conference on Supercomputing (ICS), 2011.

5.6. GGen

Participants: Guillaume Huard [correspondant], Swann Perarnau.

- Characterization of Software : A-2 / SO-4 / SM-4 / EM-2 / SDL-3
- Own Contribution: DA-4 / CD-4 / MS-4 / TPM-4
- Additional information:
  – web site: http://ggen.ligforge.imag.fr/, Coordinator Swann Perarnau
  – Objective of the software: GGen is a free (GPL-compatible) command line application and library for generating and analyzing directed acyclic graphs. Designed primarily to be used in simulations of scheduling algorithms, it helps researchers understand fully the nature of the graphs generated. It implements the most known graph generation algorithms enabling comparisons betweens them.
  – Users community: Research open source project, task scheduling community: ggen provides a meaningful way to generate test cases.
  – Positioning: To our knowledge, there's no competing tool, more details in : Daniel Cordeiro, Grégory Mounié, Swann Perarnau, Denis Trystram, Jean-Marc Vincent, and Frédéric Wagner. Random graph generation for scheduling simulations. In International ICST Conference on Simulation Tools and Techniques (SIMUTools), 2010.

5.7. Triva

Participants: Guillaume Huard [correspondant], Lucas Schnorr.

- Characterization of Software : A-2 / SO-4 / SM-5 / EM-3 / SDL-3
- Own Contribution: DA-4 / CD-3 / MS-3 / TPM-3
- Additional information:
  – web site: http://triva.gforge.inria.fr/, Coordinator, Lucas Schnorr

  - Objective of the software: Triva is an open-source tool used to analyze traces (in the pajé format) registered during the execution of parallel applications. The tool serves also as a sandbox to the development of new visualization techniques.
  - Users community: Research open source project, applications developers, especially parallel applications.
  - Positioning: Main competing tools are Vampir (classical 2D Gantt charts) and Tau (less advanced agregation techniques), more details in : A Hierarchical Aggregation Model to achieve Visualization Scalability in the analysis of Parallel Applications. Lucas Mello Schnorr, Guillaume Huard, Philippe Olivier Alexandre Navaux. Parallel Computing. Volume 38, Issue 3, March 2012.

5.8. OAR

Participants: Pierre Neyron [correspondant MOAIS], Grégory Mounié.

- Characterization of Software : A-5 / SO-3 / SM-4 / EM-4 / SDL-5
- Own Contribution: DA-3 / CD-2 / MS-1 / TPM-1
- Additional information: OAR (http://oar.imag.fr, Coordinator O. Richard, Inria MESCAL) is a batch scheduler. The MOAIS team develops the central automata and the scheduling module that includes successive evolutions and improvements of the policy.OAR is used to schedule jobs both on the CiGri (Grenoble region) and Grid50000 (France) grids. CiGri is a production grid that federates about 500 heterogeneous resources of various Grenoble laboratories to perform computations in physics. MOAIS has also developed the distributed authentication for access to Grid5000.

5.9. SOFA

Participant: Bruno Raffin [correspondant].

Inria category: ????

- Characterization of Software : A-5 / SO-4 / SM-4 / EM-4 / SDL-5
- Own Contribution: DA-2 / CD-2 / MS-1 / TPM-1
- Additional information: SOFA (http://www.sofa-framework.org/, Coordinator F. Faure, Inria IMAGINE) is an Open Source framework primarily targeted at real-time simulation, with an emphasis on medical simulation. It is mostly intended for the research community to help develop newer algorithms, but can also be used as an efficient prototyping tool. Moais contributes to parallelization of kernel algorithms used in the simulation.
- ACM: J.3
- Programming language: C/C++

5.10. LinBox

Participants: Clément Pernet [correspondant], Thierry Gautier.

- Characterization of Software : A-3 / SO-4 / SM-2 / EM-3 / SDL-5
- Own Contribution: DA-4 / CD-3 / MS-3 / TPM-4
- Additional information:
  - web site: http://linalg.org
  - Objective of the software: LinBox is an open-source C++ template library for exact, high-performance linear algebra computations. It is considered as the reference library for numerous computations (such as linear system solving, rank, characteristic polynomial, Smith normal forms,...) over finite fields and integers with dense, sparse, and structured matrices.

– The LinBox group is an international collaboration (USA: NCSU, UDel; Canada: U Waterloo, U Calgary; France: LIP, LIRMM, LJK and LIG). Articles related to the library have been published in the main Conferences of the area: ISSAC, ICMS. MOAIS contributes to its development and more specifically to its parallelization in the context of ANR HPAC project. It is currently experiencing a major change of design, to better integrate parallelism.
– Users community: mostly researchers doing computational mathematics (number theory, cryptology, group theory, persistent homology. They use the library by either linking against it directly (the library is packaged in Debian, Fedora, etc) or withing the general purpose math software Sage (sagemath.org very broad diffusion) which includes LinBox as a kernel for exact linear algebra.

ROMA Team

4. Software

4.1. MUMPS

Participants: Patrick Amestoy, Alfredo Buttari, Jean-Yves L'Excellent [correspondent], Mohamed Sid-Lakhdar, François-Henry Rouet, Bora Uçar, Clément Weisbecker.

MUMPS (for *MUltifrontal Massively Parallel Solver*, see http://graal.ens-lyon.fr/MUMPS) is a software package for the solution of large sparse systems of linear equations. The development of MUMPS was initiated by the European project PARASOL (Esprit 4, LTR project 20160, 1996-1999), whose results and developments were public domain. Since then, MUMPS has been supported by CERFACS, CNRS, ENS Lyon, INPT(ENSEEIHT)-IRIT (main contributor), Inria, and University of Bordeaux. In the context of an ADT project (Action of Technological Development), Maurice Brémond ("SED" service) also works part-time on MUMPS.

MUMPS implements a direct method, the multifrontal method; it is a parallel code capable of exploiting distributed-memory computers; its main originalities are its numerical robustness and the wide range of functionalities available.

The latest release is MUMPS 4.10.0 (May 2011).

More information on MUMPS is available at http://graal.ens-lyon.fr/MUMPS/ and http://mumps.enseeiht.fr.

RUNTIME Project-Team

5. Software

5.1. Common Communication Interface

Participant: Brice Goglin.

- The *Common Communication Interface* aims at offering a generic and portable programming interface for a wide range of networking technologies (Ethernet, InfiniBand, ...) and application needs (MPI, storage, low latency UDP, ...).
- CCI is developed in collaboration with the *Oak Ridge National Laboratory* and several other academics and industrial partners.
- CCI is in early development and currently composed of 19 000 lines of C.
- http://www.cci-forum.org

5.2. Hardware Locality

Participants: Brice Goglin, Samuel Thibault.

- *Hardware Locality* (HWLOC) is a library and set of tools aiming at discovering and exposing the topology of machines, including processors, cores, threads, shared caches, NUMA memory nodes and I/O devices.
- It builds a widely-portable abstraction of these resources and exposes it to the application so as to help them adapt their behavior to the hardware characteristics.
- HWLOC targets many types of high-performance computing applications [2], from thread scheduling to placement of MPI processes. Most existing MPI implementations, several resource managers and task schedulers already use HWLOC.
- HWLOC is developed in collaboration with the OPEN MPI project. The core development is still mostly performed by Brice GOGLIN and Samuel THIBAULT from the RUNTIME team-project, but many outside contributors are joining the effort, especially from the OPEN MPI and MPICH2 communities.
- HWLOC is composed of 40 000 lines of C.
- http://runtime.bordeaux.inria.fr/hwloc/

5.3. KNem

Participant: Brice Goglin.

- KNEM (*Kernel Nemesis*) is a Linux kernel module that offers high-performance data transfer between user-space processes.
- KNEM offers a very simple message passing interface that may be used when transferring very large messages within point-to-point or collective MPI operations between processes on the same node.
- Thanks to its kernel-based design, KNEM is able to transfer messages through a single memory copy, much faster than the usual user-space two-copy model.
- KNEM also offers the optional ability to offload memory copies on INTEL I/O AT hardware which improves throughput and reduces CPU consumption and cache pollution.
- KNEM is developed in collaboration with the MPICH2 team at the Argonne National Laboratory and the OPEN MPI project. These partners already released KNEM support as part of their MPI implementations.
- KNEM is composed of 7000 lines of C. Its main contributor is Brice GOGLIN.
- http://runtime.bordeaux.inria.fr/knem/

5.4. Marcel

Participants: Olivier Aumage, Yannick Martin, Samuel Thibault.

- MARCEL is the two-level thread scheduler (also called N:M scheduler) of the PM^2 software suite.
- The architecture of MARCEL was carefully designed to support a large number of threads and to efficiently exploit hierarchical architectures (e.g. multicore chips, NUMA machines).
- MARCEL provides a *seed* construct which can be seen as a precursor of thread. It is only when the time comes to actually run the seed that MARCEL attempts to reuse the resources and the context of another, dying thread, significantly saving management costs.
- In addition to a set of original extensions, MARCEL provides a POSIX-compliant interface which thus permits to take advantage of it by just recompiling unmodified applications or parallel programming environments (API compatibility), or even by running already-compiled binaries with the Linux NPTL ABI compatibility layer.
- For debugging purpose, a trace of the scheduling events can be recorded and used after execution for generating an animated movie showing a replay of the execution.
- The MARCEL thread scheduling library is made of 80 000 lines of code.
- http://runtime.bordeaux.inria.fr/marcel/
- Marcel has been supported for 2 years (2009-2011) by the Inria ADT Visimar.

5.5. ForestGOMP

Participants: Olivier Aumage, Yannick Martin, Pierre-André Wacrenier.

- FORESTGOMP is an OPENMP environment based on both the GNU OPENMP run-time and the MARCEL thread library.
- It is designed to schedule efficiently nested sets of threads (derived from nested parallel regions) over hierarchical architectures so as to minimize cache misses and NUMA penalties.
- The FORESTGOMP runtime generates nested MARCEL bubbles each time an OPENMP parallel region is encountered, thereby grouping threads sharing common data.
- Topology-aware scheduling policies implemented by BUBBLESCHED can then be used to dynamically map bubbles onto the various levels of the underlying hierarchical architecture.
- FORESTGOMP allowed us to validate the BUBBLESCHED approach with highly irregular, fine grain, divide-and-conquer parallel applications.
- http://runtime.bordeaux.inria.fr/forestgomp/

5.6. Open-MX

Participant: Brice Goglin.

- The OPEN-MX software stack is a high-performance message passing implementation for any generic ETHERNET interface.
- It was developed within our collaboration with Myricom, Inc. as a part of the move towards the convergence between high-speed interconnects and generic networks.
- OPEN-MX exposes the raw ETHERNET performance at the application level through a pure message passing protocol.
- While the goal is similar to the old GAMMA stack [35] or the recent iWarp [34] implementations, OPEN-MX relies on generic hardware and drivers and has been designed for message passing.
- OPEN-MX is also wire-compatible with Myricom MX protocol and interface so that any application built for MX may run on any machine without Myricom hardware and talk other nodes running with or without the native MX stack.
- OPEN-MX is also an interesting framework for studying next-generation hardware features that could help ETHERNET hardware become legacy in the context of high-performance computing. Some innovative message-passing-aware stateless abilities, such as multiqueue binding and interrupt coalescing, were designed and evaluated thanks to OPEN-MX [5].
- Brice GOGLIN is the main contributor to OPEN-MX. The software is already composed of more than 45 000 lines of code in the Linux kernel and in user-space.
- http://open-mx.org/

5.7. StarPU

Participants: Cédric Augonnet, Olivier Aumage, Nicolas Collin, Nathalie Furmento, Cyril Roelandt, Ludovic Stordeur, Samuel Thibault, Ludovic Courtès.

- STARPU permits high performance libraries or compiler environments to exploit heterogeneous multicore machines possibly equipped with GPGPUs or Cell processors.
- STARPU offers a unified offloadable task abstraction named codelet.In case a codelet may run on heterogeneous architectures, it is possible to specify one function for each architectures (e.g. one function for CUDA and one function for CPUs).
- STARPU takes care to schedule and execute those codelets as efficiently as possible over the entire machine. A high-level data management library enforces memory coherency over the machine: before a codelet starts (e.g. on an accelerator), all its data are transparently made available on the compute resource.
- STARPU obtains portable performances by efficiently (and easily) using all computing resources at the same time.
- STARPU also takes advantage of the heterogeneous nature of a machine, for instance by using scheduling strategies based on auto-tuned performance models.
- STARPU can also leverage existing parallel implementations, by supporting *parallel tasks*, which can be run concurrently over the machine.
- STARPU provides a *reduction* mode, which permit to further optimize data management when results have to be reduced.
- STARPU provides integration in MPI clusters through a lightweight DSM over MPI.
- STARPU comes with a plug-in for the GNU Compiler Collection (GCC), which extends languages of the C family with syntactic devices to describe STARPU's main programming concepts in a concise, high-level way.
- http://runtime.bordeaux.inria.fr/StarPU/

5.8. NewMadeleine

Participants: Alexandre Denis, François Trahay, Raymond Namyst.

- NEWMADELEINE is communication library for high performance networks, based on a modular architecture using software components.
- The NEWMADELEINE optimizing scheduler aims at enabling the use of a much wider range of communication flow optimization techniques such as packet reordering or cross-flow packet aggregation.
- NEWMADELEINE targets applications with irregular, multiflow communication schemes such as found in the increasingly common application conglomerates made of multiple programming environments and coupled pieces of code, for instance.
- It is designed to be programmable through the concepts of optimization *strategies*, allowing experimentations with multiple approaches or on multiple issues with regard to processing communication flows, based on basic communication flows operations such as packet merging or reordering.
- The reference software development branch of the NEWMADELEINE software consists in 90 000 lines of code. NEWMADELEINE is available on various networking technologies: Myrinet, Infiniband, Quadrics and ETHERNET. It is developed and maintained by Alexandre DENIS.
- http://runtime.bordeaux.inria.fr/newmadeleine/

5.9. PadicoTM

Participant: Alexandre Denis.

- PadicoTM is a high-performance communication framework for grids. It is designed to enable various middleware systems (such as CORBA, MPI, SOAP, JVM, DSM, etc.) to utilize the networking technologies found on grids.
- PadicoTM aims at decoupling middleware systems from the various networking resources to reach transparent portability and flexibility.
- PadicoTM architecture is based on software components. Puk (the PadicoTM micro-kernel) implements a light-weight high-performance component model that is used to build communication stacks.
- PadicoTM component model is now used in NEWMADELEINE. It is the cornerstone for networking integration in the projects “LEGO” and “COOP” from the ANR.
- PadicoTM is composed of roughly 60 000 lines of C.
- PadicoTM is registered at the APP under number IDDN.FR.001.260013.000.S.P.2002.000.10000.
- http://runtime.bordeaux.inria.fr/PadicoTM/

5.10. MAQAO

Participants: Denis Barthou, Andres Charif-Rubial.

- MAQAO is a performance tuning tool for OpenMP parallel applications. It relies on the static analysis of binary codes and the collection of dynamic information (such as memory traces). It provides hints to the user about performance bottlenecks and possible workarounds.
- MAQAO relies on binary codes and inserts probes for instrumention directly inside the binary. There is no need to recompile. The static/dynamic approach of MAQAO analysis is the main originality of the tool, combining performance model with values collected through instrumentation.
- MAQAO has a static performance model for x86 architecture and Itanium. This model analyzes performance of the predecoder, of the decoder and of the different pipelines of the x86 architecture, in particular for SSE instructions.
- The dynamic collection of data in MAQAO enables the analysis of thread interactions, such as false sharing, amount of data reuse, runtime scheduling policy, ...
- MAQAO is in the project ”ProHMPT” from the ANR. A demo of MAQAO has been made in Jan. 2010 for SME/Inria days and in Nov. 2010 at SuperComputing, Inria Booth.
- http://www.maqao.org/

5.11. QIRAL

Participant: Denis Barthou.

- QIRAL is a high level language (expressed through LaTeX) that is used to described Lattice QCD problems. It describes matrix formulations, domain specific properties on preconditionings, and algorithms.
- The compiler chain for QIRAL can combine algorithms and preconditionings, checking validity of the composition automatically. It generates OpenMP parallel code, using libraries, such as BLAS.
- This code is developped in collaboration with other teams participating to the ANR PetaQCD project.

5.12. TreeMatch

Participants: Emmanuel Jeannot, Guillaume Mercier, François Tessier.

- TREEMATCH is a library for performing process placement based on the topology of the machine and the communication pattern of the application.
- TREEMATCH provides a permutation of the processes to the processors/cores in order to minimize the communication cost of the application.
- Important features are : the number of processors can be greater than the number of applications processes ; it assumes that the topology is a tree and does not require valuation of the topology (e.g. communication speeds) ; it implements different placement algorithms that are switched according to the input size.
- TREEMATCH is integrated into various software such as the Charm++ programming environment as well as in both major open-source MPI implementations: Open MPI and MPICH2.
- TREEMATCH is available at: http://treematch.gforge.inria.fr.

DANTE Team

5. Software

5.1. Sensor Network Tools: drivers, OS and more

Participants: Éric Fleury [correspondant], Sandrine Avakian.

As a outcomes of the ANR SensLAB project and the Inria ADT SensTOOLS and SensAS, several softwares (from low level drivers to OSes) were delivered and made available to the research community. The main goal is to lower the cost of developing/deploying a large scale wireless sensor network application. All software are gathered under the SensLAB web site: http://www.senslab.info/ web page where one can find:

- low C-level drivers to all hardware components;
- ports of the main OS, mainly TinyOS, FreeRTOS and Contiki;
- ports and development of higher level library like routing, localization.

5.2. http://queueing-systems.ens-lyon.fr

Participant: Thomas Begin [correspondant].

Queueing models, steady-state solution, online tool, web interface

This tool aims at providing a simple web based interface to promote the use of our proposed solutions to numerically solve classical queueing systems. In 2011, the tools merely implemented the solution to get the distribution for the number of customers along with customary performance parameters for a queue with multiple servers, general arrivals, exponential services and a possibly finite buffer, (i.e., $Ph/M/c/N$-like queue). The steady-state solution to this queue is based on a simple and stable recurrence [2] and was performed in collaboration with Pr. Brandwajn (UCSC). In 2012 we extended our tool so as to include the solution for a queue with a single server, Poisson arrivals, general services and a possibly finite buffer, (i.e., $M/Ph/1/N$-like queue). Our tool was presented at the conference [43] and attracts hundreds of visitors each month. Associated URL is: http://queueing-systems.ens-lyon.fr

DIONYSOS Project-Team

4. Software

4.1. T3devKit testing toolkit and IPv6 test suites

Participants: Anthony Baire, César Viho.

We have built a toolkit for easing executing tests written in the standardized TTCN-3 test specification language. This toolkit is made of a C++ library together with a highly customizable CoDec generator that allows fast development of external components (that are required to execute a test suite) such as CoDec (for message Coding/Decoding), System and Platform Adapters. It also provides a framework for representing and manipulating TTCN-3 events so as to ease the production of test reports. The toolkit addresses issues that are not yet covered by ETSI standards while being fully compatible with the existing standard interfaces: TRI (Test Runtime Interfaces) and TCI (Test Control Interfaces), it has been tested with four TTCN-3 environments (IBM, Elvior, Danet and Go4IT) and on three different platforms (Linux, Windows and Cygwin). It is publicly released under the CeCILL-C License.

All these tools with associated test suites (for RIPng, DHCPv6 and examples for DNS) are freely available at http://www.irisa.fr/tipi.

4.2. Interoperability Assessment

Participants: Anthony Baire, Nanxing Chen, Arulnambi Nandagoban, César Viho.

Our experience in interoperability assessment (since 1996) and in using the TTCN-3 standard allowed us to develop a tool (called `ttproto`) that helps in: (i) experimenting new concepts for long term evolution of the TTCN-3 standard [37] and (ii) facilitating new approaches and methods for interoperability assessment. For instance, new passive approaches (see [45], [46], [47]) that we developed have been implemented and validated using `ttproto`. This tool `ttproto` has been used to develop test suites for 6LoWPAN-ND (IPv6 for Low Power Networks) and CoAP (Constrained Application Protocol). The CoAP test suites have been successfully used for two Plugtest interoperability events organized by ETSI, IPSO Alliance and the FP7 PROBE-IT project, respectively 28-29 March in Paris (see [44]) and 28-30 November in Sophia-Antipolis. The tool `ttproto` and the test suites indicated above are freely available at http://www.irisa.fr/tipi.

4.3. Performance and dependability evaluation

Participants: Gerardo Rubino, Bruno Sericola, Bruno Tuffin.

We develop software tools for the evaluation of two classes of models: Markov models and reliability networks. The main objective is to quantify dependability aspects of the behaviors of the modeled systems, but other aspects of the systems can be handled (performance, performability, vulnerability). The tools are specialized libraries implementing numerical, Monte Carlo and Quasi-Monte Carlo algorithms.

One of these libraries has been developed for the Celar (DGA), and its goal is the evaluation of dependability and vulnerability metrics of wide area communication networks (WANs). The algorithms in this library can also evaluate the sensitivities of the implemented dependability measures with respect to the parameters characterizing the behavior of the components of the networks (nodes, lines).

We are also developing tools with the objective of building Markovian models and to compute bounds of asymptotic metrics such as the asymptotic availability of standard metrics of models in equilibrium, loss probabilities, blocking probabilities, mean backlogs,...). A set of functions designed for dependability analysis is being built under the name `DependLib`.

DISTRIBCOM Project-Team

5. Software

5.1. SOFAT

Participants: Loïc Hélouët [correspondant], Rouwaida Abdallah.

SOFAT is the acronym for Scenario Oracle and Formal Analysis Toolbox. As this name suggests it is a formal analysis toolbox for scenarios. Scenarios are informal descriptions of behaviors of distributed systems. SOFAT allows the edition and analysis of distributed systems specifications described using Message Sequence Charts, a scenario language standardized by the ITU [Z.120]. The main functionalities proposed by SOFAT are the textual edition of Message Sequence Charts, their graphical visualization, the analysis of their formal properties, and their simulation. The analysis of the formal properties of a Message Sequence Chart specification determines if a description is regular, local choice, or globally cooperative. Satisfaction of these properties allow respectively for model-checking of logical formulae in temporal logic, implementation, or comparison of specifications. All these applications are either undecidable problems or unfeasible if the Message Sequence Chart description does not satisfy the corresponding property. The SOFAT toolbox implements most of the theoretical results obtained on Message Sequence Charts this last decade. It is regularly updated and re-distributed. The purpose of this is twofold:

- Provide a scenario based specification tool for developers of distributed applications
- Serve as a platform for theoretical results on scenarios and partial orders

SOFAT provides several functionalities, that are: syntactical analysis of scenario descriptions, Formal analysis of scenario properties, Interactive Simulation of scenarios when possible, and diagnosis. This year, SOFAT was extended with code synthesis functionalities, allowing to generate communicating automata, promela code, or rest based web services from HMSCs. A new release of the software is expected before the end of the year.

See also the web page http://www.irisa.fr/distribcom/Prototypes/SOFAT/index.html.

- AMS: Order; lattices; ordered algebraic structures
- APP: IDDN.FR.001.080027.000.S.P.2003.00.10600
- Programming language: Java

5.2. PLASMA

Participants: Sean Sedwards, Benoit Boyer, Kevin Corre, Axel Legay [correspondant].

PLASMA is our implementation of Statistical Model Checking. PLASMA adopts a modular architecture to facilitate the extension of its features. Models can currently be specified using the PRISM reactive modules syntax or a biochemical syntax, while properties are specified in a discrete bounded temporal logic. Our goal is to allow the implementation of other modeling languages and logics by means of self-contained *drop-in* modules. PLASMA facilitates this by providing an intermediate language to generate transition systems based on the notion of the construct (guard, rate, actions), where guard, rate and actions are functions over the current state of the system and control whether and how fast the system may perform certain actions in each state. New modeling languages may be thus added to PLASMA's repertoire by constructing parsers that translate such languages into the intermediate language.

Web site: https://project.inria.fr/plasma-lab/

5.3. LotrecScheme

Participant: François Schwarzentruber [correspondant].

LotrecScheme is the implementation of a generic tableau method prover based on LoTREC (http://www.irit.fr/Lotrec/). LotrecScheme is more expressive than LoTREC. Both LoTREC and LotrecScheme provides tableau methods for standard modal logic K, KT, S4, etc. Contrary to LoTREC, LotrecScheme is expressive enough to capture some satisfiability problem for Dynamic Epistemic Logic.

The prover inside LotrecScheme is written in Scheme and embedded in a JAVA application.

See also the web page http://www.irisa.fr/distribcom/Prototypes/LotrecScheme/index.html.

FUN Team

4. Software

4.1. Distributed ONS

Participants: Nathalie Mitton, Roberto Quilez [correspondant].

This module implements a DHT-based Distributed EPC Global ONS issued from the ANR WINGS project and published in [30]. APP number: IDDN.FR.001.180033.000.S.P.2012.000.10000.

- Version: version 1

4.2. GOLIATH 1.0

Participants: Fadila Khadar [correspondant], Nathalie Mitton.

GOLIATH (Generic Optimized LIghtweight communication stack for Ambient TecHnologies) is a full protocol stack for wireless sensor networks.

See also the web page https://gforge.inria.fr/projects/goliath/.

4.3. Linear variable energy module for WSNET.

Participants: Tony Ducrocq [correspondant], Nathalie Mitton.

This module is to be integrated in the WSNET event-based simulator for wireless networks. It implements a Linear transmission variable energy module for WSNET.

- Version: 1.0

GANG Project-Team (section vide)

HIPERCOM Project-Team

5. Software

5.1. RPL P2P

Participants: Emmanuel Baccelli [correspondant], Oliver Hahm, 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. MPR-OSPF

Participants: Emmanuel Baccelli [correspondant], Juan-Antonio Cordero.

MPR-OSPF is an implementation of RFC5449, providing OSPF-compatible routing in hybrid networks composed of both mobile ad hoc routers and fixed wired networks. The implementation is based on Quagga/Zebra. See also the web page http://ospfmanet.gforge.inria.fr.

- Version: 0.4

5.3. OPERA infrastructure

Participants: Cédric Adjih [correspondant], Ichrak 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.

5.4. OPERA perf simul

Participants: Cédric Adjih [correspondant], Ichrak 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.

5.5. OPERA protocol

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

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.

5.6. 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.

MADYNES Project-Team

5. Software

5.1. SecSIP

Participants: Abdelkader Lahmadi [contact], Olivier Festor.

SecSip[1] is developed by the team to defend SIP-based (The Session Initiation Protocol) services from known vulnerabilities. It presents a proactive point of defense between a SIP-based network of devices (servers, proxies, user agents) and the open Internet. Therefore, all SIP traffic is inspected and analyzed against authored Veto specification before it is forwarded to these devices. When initializing, the SecSIP runtime starts loading and parsing authored VeTo blocks to identify different variables, event patterns, operations and actions from each rule. Veto is a generic declarative language for attack patterns specification. SecSIP implements an input and output layer, to capture, inject, send and receive SIP packets from and to the network. Intercepted packets are moved to the SIP Packet parser module. The main function of this module is to extract different fields within a SIP message and trigger events specified within the definition blocks. During each execution cycle when a SIP message arrives, the SecSIP runtime uses a data flow acyclic graph network to find definition matching rules and triggers defined events. The paired events in each operator node are propagated over the graph until a pattern is satisfied. When the pattern is satisfied, the respective rule is fired and the set of actions is executed.

SecSIP is freely available on the Internet. It was extended to support new protocols in the area of SCADA systems in 2012.

5.2. NDPMon

Participants: Isabelle Chrisment, Olivier Festor [contact].

The Neighbor Discovery Protocol Monitor (NDPMon) is an IPv6 implemention of the well-known ArpWatch tool. NDPMon monitors the pairing between IPv6 and Ethernet addresses (NDP activities: new station, changed Ethernet address, flip flop...). NDPMon also detects attacks on the NDP protocol, as defined in RFC 3756 (bogon, fake Router Advertisements...). New attacks based on the Neighbor Discovery Protocol and Address Auto-configuration (RFC 2461 and RFC 2462) have been identified and integrated in the tool. An XML file describes the default behavior of the network, with the authorized routers and prefixes, and a second XML document containing the neighbors database is used. This second file can be filled during a learning phase. All NDP activities are logged in the syslog utility, and so the attacks, but these ones are also reported by mail to the administrator. Finally, NDPMon can detect stack vulnerabilities, like the assignment of an Ethernet broadcast address on an interface.

NDPMon comes along with a WEB interface acting as a GUI to display the informations gathered by the tool, and give an overview of all alerts and reports. Thanks to color codes, the WEB interface makes possible for the administrator to have an history of what happened on his network and identify quickly problems. All the XML files used or produced by the daemon (neighbor cache, configuration file and alerts list) are translated in HTML via XSL for better readability. A statistic module is also integrated and gives informations about the discovery of the nodes and their type (MAC manufacturer distribution ...).

The software package and its source code is freely distributed under an opensource license (LGPL). It is implemented in C, and is available through a SourceForge project at http://ndpmon.sf.net. An open source community is now established for the tool which has distributions for several Operating Systems (Linux, FreeBSD, OpenBSD, NetBSD and Mac OS X). It is also integrated in FreeBSD ports at http://www.freebsd.org/cgi/cvsweb.cgi/ports/net-mgmt/ndpmon/. Binary distributions are also available for .deb and .rpm based Linux flavors.

In 2012, the software underwent a complete reshaping thanks to a substantial support from the High Security Lab which dedicated us 6 months of research engineer.

[1] http://secsip.gforge.inria.fr/doku.php

MAESTRO Project-Team (section vide)

MASCOTTE Project-Team

5. Software

5.1. Grph

Participants: David Coudert, Luc Hogie [correspondant], Aurélien Lancin, Grégory Morel, Issam Tahiri.

Around 20,000 lines of code, developed in Java.

The objective of GRPH is to provide researchers and engineers a suitable graph library for graph algorithms experimentation and network simulation. GRPH is primarily a software library, but it also comes with a set of executable files for user interaction and graph format conversion; as such, it can be used autonomously. Performance and accessibility are the primary targets of the GRPH library. It allows manipulating large graphs (millions of nodes). Its model considers mixed graphs composed of directed and undirected simple- and hyper-edges. GRPH comes with a collection of base graph algorithms which are regularly augmented.

So far, known users of the GRPH library include people at Mascotte and others involved in the FP7 EULER project. It got some contribution from the Inria team GANG who contributed GRPH with an implementation of the four-sweep algorithm which provides accurate lower bound on the diameter in linear time. It has a number of other academic users including research students at Bergamo University (Italy), and University of Southern Denmark (students supervised by Jørgen Bang-Jensen).

GRPH includes bridges to other graph libraries such as JUNG, JGraphT, CORESE (a software developed by the WIMMICS team Inria-I3S), LAD (Christine Solnon, LIRIS), Nauty (Brendan D. McKay), as well as specific algorithms developed by Matthieu Latapy and Jean-Lou Guillaume (LIP6), etc.

GRPH is distributed under the terms of a license defined by its contributors and is available for download. This license allows free usage and access to the source code. See http://www-sop.inria.fr/mascotte/software/grph.

In 2012, numerous graph algorithms have been added to GRPH, such as maximum matching, minimum vertex cover (brute force, branching, Niedermeier), maximum independent set (Fomin/Grandoni/Kratsch). Furthermore, to answer a number of issues about the generation of graph instances with particular properties, a framework for evolutionary computing dedicated to graphs was integrated to GRPH. Moreover, a reworked version of Mascsim was integrated in GRPH.

On-going works concern the distributed execution of graph algorithms, and a bridge to Sage.

See also the web page http://www-sop.inria.fr/mascotte/software/grph/.

5.2. DRMSim

Participants: David Coudert, Luc Hogie [correspondant], Aurélien Lancin, Nicolas Nisse, Issam Tahiri.

Around 45,000 lines, developed in Java, collaboration between MASCOTTE and LaBRI.

DRMSim relies on a discrete-event simulation engine aiming at enabling the large-scale simulations of routing models. DRMSim is developped in the framework of the FP7 EULER project . It proposes a general routing model which accommodates any network configuration. Aside to this, it includes specific models for Generalized Linear Preference (GLP), and k-chordal network topologies, as well as implementations of routing protocols, including the routing protocol proposed in [37] and lightweight versions of BGP (Border Gateway Protocol).

The system model considers the dynamic evolution of the simulated network. This model takes as its input parameter the distribution of failure probability for both routers and links.

The metric model takes measures along a discrete-event simulation which can be performed in many ways.

Commonly, a simulation campaign consists in iterating over the set of combinations of parameter values, calling the simulation function for every combination. These combinations are most often complex, impeding there description by a set of mathematical functions. Thus DRMSim provides a simulation methodology that describes (programmatically) the way a simulation campaign should be conducted.

DRMSim stores on disk every step of the execution of a simulation campaign. In a simulation campaign, simulation runs are independent (no simulation depends on the result computed by another simulation). Consequently they can be executed in parallel. Because one simulation is most likely to use large amount of memory and to be multi-threaded, parallelizing the simulation campaign on one single computer is a poor parallelization scheme. Instead, we currently work at enabling the remote parallel execution of several simulation runs, with the same distribution framework that is used in the GRPH library.

DRMSim relies on the Mascsim abstract discrete-event simulation framework, the GRPH library and the Java4Unix integration framework.

Finally, from an object-oriented point of view of its conception model, DRMSim manipulates graph abstractions, allowing the user to force the use of a library different from the default one, i.e. GRPH.

See also the web page http://www-sop.inria.fr/mascotte/projets/DCR/.

5.3. SageMath

Participants: David Coudert, Leonardo Sampaio.

Developed in Python, Cython, and C++. MASCOTTE members have already contributed to the development of more than 180 patches and to the reviewing process of more than 200 patches that are now part of the standard distribution.

Sagemath is a free open-source mathematics software aiming at becoming an alternative to Maple and Matlab. Initially created by William Stein (Professor of mathematics at Washington University), Sagemath is currently developed by more than 180 contributors around the world (mostly researchers) and its source code has reached 350 MB. It is of interest for Mascotte members because it combines a large collection of graph algorithms with various libraries in algebra, calculus, combinatorics, linear programming, statistics, etc.

We use Sagemath for quickly testing algorithms, analyzing graphs, and disseminating algorithms. We also use it for teaching purposes in the Master IFI, stream UBINET.

In 2012, David Coudert has contributed to the development of the Sage releases 5.0 to 5.6 with 15 patches (from bug fix to advance graph algorithms) and participated to the reviewing process of more than 30 patches.

5.4. Utilities

5.4.1. Java4unix

Participant: Luc Hogie [correspondant].

More than 5,000 lines, developed in Java.

Java4unix proposes a development and distribution framework which simplifies the use of Java for UNIX software programming/distribution. Until now, Java could hardly be used for the development UNIX applications because invoking Java applications from the UNIX shell must be done through an explicit call to the Java virtual machine and writing simple things in Java often requires long coding. Java4unix aims at filling those two gaps by providing a UNIX installer for java applications, turning them to standard UNIX application and a framework that UNIX programmers may use to manipulate files/text, etc.

Java4unix includes a module which enables the reporting and automatic releasing of Eclipse Java projects.

See also the web page http://www-sop.inria.fr/members/Luc.Hogie/java4unix/.

5.4.2. Jalinopt

Participants: Luc Hogie [correspondant], Grégory Morel.

Developed in Java.

Jalinopt is a Java toolkit for building and solving linear programs. It consists of a straightforward object-oriented model for linear programs, as well as a bridge to most common solvers, including GLPK and CPLEX. It is an interface to many LP solvers allowing users to code independently of the solver effectively. Altought Jalinopt is inspired by Mascopt and JavaILP, it provides a significantly different model and an utterly different approach to connecting to the solver. In particular this approach, based in inter-process piping, offers better portability, and the possibility to connect (via SSH) to solvers on remote computers.

In 2012, we refined the object-oriented model of Jalinopt and improved its portability by making it working with LPSolve as its default native solver.

See also the web page http://www-sop.inria.fr/members/Luc.Hogie/jalinopt/.

5.4.3. JavaFarm

Participant: Luc Hogie [correspondant].

More than 1,500 lines, developed in Java.

JavaFarm is a middleware enabling the distribution of Java applications across farms of servers.

Its workflow basically enables an application to locally aggregate code and data into an object, called job, that will migrate to another computer where it will be computed. When a job completes, its result is transferred back to the caller. Among other features, JavaFarm supports futures (asynchronous job executions), thereby enabling parallelization of the distributed code. The design objectives of JavaFarm are to make distribution and parallelism as transparent and easy as possible.

See also the web page http://www-sop.inria.fr/members/Luc.Hogie/javafarm/.

5.4.4. Mascsim

Participants: Luc Hogie [correspondant], Aurélien Lancin, Issam Tahiri.

Around 12,000 lines, developed in Java.

Mascsim is a distributed discrete event simulator whose main target is to be easy to use. Unlike most discrete-event simulators, the researcher who is using Mascsim is required to provide only the bare minimum material needed for the simulation: a model for the system, a set of events describing what is going on in the system, as well as a set of metrics of interest. The simulation process is then entirely automatized.

In 2012, Mascsim was adapted and integrated to GRPH.

See also the web page http://www-sop.inria.fr/mascotte/software/mascsim/.

5.4.5. P2PVSim

Participant: Remigiusz Modrzejewski [correspondant].

Around 12,000 lines, developed in Python.

P2PVSim is a simple discrete-event simulator created for analyzing theoretical properties of peer-to-peer live video streaming algorithms. Implemented in Python it was designed with clarity and extensibility in mind from the beginning. It is capable of simulating overlays of a few thousands of peers. Multiple control protocols have been implemented. At the same time, a lot of work was put into the performance and scalability aspects of the software. Currently it is meant for simulating overlays of a few thousand peers running multiple control protocols that have been implemented.

In 2012, a distributed version of P2PVSim was developed. The objectives for developing a distributed version was to fasten the simulation of large campaigns, that would be too long to run on one single computer. The distributed P2PVSim runs on an arbitrary number of computers. It has been so far used with success on a dozen computers with multiple cores all located in the same LAN.

PLANETE Project-Team

5. Software

5.1. ns-3

Participant: Daniel Camara [correspondant].

ns-3 is a discrete-event network simulator for Internet systems, targeted primarily for research and educational use. ns-3 is free software, licensed under the GNU GPLv2 license, and is publicly available for research, development, and use. ns-3 includes a solid event-driven simulation core as well as an object framework focused on simulation configuration and event tracing, a set of solid 802.11 MAC and PHY models, an IPv4, UDP, and TCP stack and support for nsc (integration of Linux and BSD TCP/IP network stacks).
See also the web page http://www.nsnam.org.

- Version: ns-3.7
- Keywords: networking event-driven simulation
- License: GPL (GPLv2)
- Type of human computer interaction: programmation C++/python, No GUI
- OS/Middleware: Linux, cygwin, osX
- Required library or software: standard C++ library: GPLv2
- Programming language: C++, python
- Documentation: doxygen

5.2. EphPub

Participants: Mohamed Ali Kaafar [correspondant], Claude Castelluccia.

EphPub (Ephemeral Publishing) (previously called EphCom) implements a novel key storage mechanism for time-bounded content, that relies on the caching mechanism of the Domain Name System (DNS). Features of EphPub include: EphPub exploits the fact that DNS servers temporarily cache the response to a recursive DNS query for potential further requests. EphPub provides higher security than Vanish, as it is immune to Sybil attacks. EphPub is easily deployable and does not require any additional infrastructure, such as Distributed Hash Tables. EphPub comes with high usability as it does not require users to install and execute any extra additional software. EphPub lets users define data lifetime with high granularity. We provide EphPub as an Android Application to provide ephemeral exchanged SMS, emails, etc. and as a Firefox or Thunderbird extensions so as to support ephemeral publication of any online document.
For more details about the different software products, see http://planete.inrialpes.fr/projects/ephemeral-publication/.

- Version: v0.1.2-beta
- ACM: K.4.1
- AMS: 94Axx
- Keywords: Ephemeral communications, Right to Forget, Future Internet Architecture, Privacy
- Software benefit: We provide a Firefox Extension that easily allows users to manage disappearing emails. We also provide a command-line tool to manage disappearing files.
- APP: Under APP deposit internal process
- License: GPL
- Type of human computer interaction: Firefox extension + Unix Console
- OS/Middelware: Firefox under any OS
- Required library or software: Python Ext
- Programming language: Python
- Documentation: No detailed documentation has been released so far. A detailed howto can be consulted however at: http://code.google.com/p/disappearingdata/source/browse/wiki/EphCOM_Firefox_Extension.wiki?r=77

5.3. Username Tester

Participants: Claude Castelluccia [correspondant], Mohamed Ali Kaafar, Daniele Perito.

Usernames are ubiquitous on the Internet. Almost every web site uses them to identify its users and, by design, they are unique within each service. In web services that have millions or hundreds of millions of users, it might become difficult to find a username that has not already been taken. For instance, you might have experienced that a specific username you wanted was already taken. This phenomenon drives users to choose increasingly complex and unique usernames.

We built a tool to estimate how unique and linkable usernames are and made it available on this page for you to check. For example, according to our tool, "ladygaga" or "12345678″ only carry 24 and 17 bits of entropy, respectively. They are therefore not likely to be unique on the Internet. On the other hand, usernames such as "pdjkwerl" or "yourejerky" carry about 40 bits of entropy and are therefore very good identifiers.

Type your username (for example "zorro1982″ or "dan.perito") to discover how unique it is. This tool can help you to select an username that has low entropy and can't be used to track you on the Internet.

Alternatively, try typing two usernames separated by a space. The tool will give an estimation on whether the two usernames are linkable. The tool is accessible here: http://planete.inrialpes.fr/projects/how-unique-are-your-usernames/

5.4. DroidMonitor

Participants: Claude Castelluccia [correspondant], Mohamed Ali Kaafar.

In nowadays world the technological progress evolves very quickly. There are more and more new devices, fully equipped with the latest innovations. The question is: do we adopt our main privacy concerns according to these new technologies as quickly as they grow and become widely available for us?...

We developed a novel tool, private data leakage monitoring tool, DroidMonitor. It aims to serve as an educational tool for regular Android Smartphones users to make them aware of existing privacy threats while they are using Location-Based Services. It can be downloaded here: http://planete.inrialpes.fr/android-privacy/

5.5. NEPI

Participants: Thierry Turletti [correspondant], Alina Quereilhac.

NEPI stands for Network Experimentation Programming Interface. NEPI implements a new experiment plane used to perform ns-3 simulations, planetlab and emulation experiments, and, more generally, any experimentation tool used for networking research. Its goal is to make it easier for experimenters to describe the network topology and the configuration parameters, to specify trace collection information, to deploy and monitor experiments, and, finally, collect experiment trace data into a central datastore. NEPI is a python API (with an implementation of that API) to perform all the above-mentioned tasks and allows users to access these features through a simple yet powerful graphical user interface called NEF.

During the year 2012 we improved support for PlanetLab experiments in NEPI, adding the ability to create customized routing overlays on top of PlanetLab. Details on these improvements can be found in [48]. We also included the ability to easily conduct CCNx http://www.ccnx.org/ experiments using PlanetLab nodes. This work was presented at the CCNx 2012 community meeting [73], and has had a good impact on the number of NEPI users.

Additionally, ongoing work on the context of the Openlab, Fed4Fire and Simulbed projects, has lead to a number of interesting extensions to NEPI. We are currently developing support to conduct experiments on OMF wireless testbeds (http://mytestbed.net/). We are also working to support DCE enabled experimentation, using the ns-3 simulator, in NEPI. Furthermore, recent work on improving NEPI's experiment control architecture, to enable both easier extension to new experimentation platforms and improve the user ability to control of experiment tasks, was presented at the CoNEXT'12 Students Workshop (see [61]).

For more information, see also the web page http://nepi.inria.fr.

- Version: 2.0
- ACM: C.2.2, C.2.4
- Keywords: networking experimentation
- License: GPL (2)
- Type of human computer interaction: python library, QT GUI
- OS/Middelware: Linux
- Required library or software: python – http://www.python.org – http://rpyc.sourceforge.net
- Programming language: python

5.6. Reference implementation for SFA Federation of experimental testbeds

Participants: Thierry Parmentelat [correspondant], Julien Tribino.

We are codevelopping with Princeton University a reference implementation for the Testbed-Federation architecture known as SFA for Slice-based Federation Architecture. During 2011 we have focused on the maturation of the SFA codebase, with several objectives in mind, better interoperabity between the PlanetLab world and the EmuLab, a more generic shelter that other testbeds can easily leverage in order to come up with their own SFA-compliant wrapper and support for 'reservable' mode, which breaks the usual best-effort PlanetLab model. For more details about this contribution see section

See also the web page http://planet-lab.eu

- Version: myplc-5.0-rc26
- Keywords: networking testbed virtual machines
- License: Various Open Source Licences
- Type of human computer interaction: Web-UI, XMLRPC-based API, Qt-based graphical client
- OS/Middelware: Linux-Fedora
- Required library or software: Fedora-14 for the infrastructure side; the software comes with a complete software suite for the testbed nodes
- Programming languages: primarily python, C, ocaml
- Documentation: most crucial module plcapi is self-documented using a local format & related tool. See e.g. https://www.planet-lab.eu/db/doc/PLCAPI.php
- Codebase: http://git.onelab.eu

5.7. SfaWrap

Participants: Thierry Parmentelat [correspondant], Mohamed Larabi.

The SfaWrap is a reference implementation of the Slice-based Federation Architecture (SFA), the emerging standard for networking experimental testbed federation. We are codeveloping the SfaWrap with Princeton University, and during 2012, we have focused on:

- Participating in the discussions about the future and evolutions of the architecture of SFA, as part of the architecture working group of the GENI project.
- Turning this initially Planet-Lab specific implementation into a generic one, that testbed providers can easily leverage for bringing SFA-compliance to their own testbeds.
- Supporting the allocation and provisioning of both 'Exclusive' and 'Shared' testbed resources.
- Enlarging the federation scheme by federating various testbeds with heterogeneous resources, in order to allow researchers to combine all available resources and run advanced networking experiments of significant scale and diversity.

- Version: sfa-2.1-22, myplc-5.0-rc33
- Keywords: networking testbed federation
- License: Various Open Source Licenses
- Type of human computer interaction: Web-UI, XMLRPC-based API, Qt-based graphical client
- OS/Middelware: Linux
- Required library or software: python2.5 or superior
- Programming languages: python
- Documentation: http://svn.planet-lab.org/#SFAUser-leveldocumentation
- Codebase: http://git.onelab.eu/?p=sfa.git;a=summary

5.8. MultiCast Library Version 3

Participant: Vincent Roca [correspondant].

MultiCast Library Version 3 is an implementation of the ALC (Asynchronous Layered Coding) and NORM (NACK-Oriented Reliable Multicast Protocol) content delivery Protocols, and of the FLUTE/ALC file transfer application. This software is an implementation of the large scale content distribution protocols standardized by the RMT (Reliable Multicast Transport) IETF working group and adopted by several standardization organizations, in particular 3GPP for the MBMS (Multimedia Broadcast/Multicast Service), and DVB for the CBMS (Convergence of Broadcast and Mobile Services). Our software is used in operational, commercial environments, essentially in the satellite broadcasting area and for file delivery over the DVB-H system where FLUTE/ALC has become a key component. See http://planete-bcast.inrialpes.fr/ for more information.

5.9. OpenFEC.org: because open, free AL-FEC codes and codecs matter

Participants: Vincent Roca [correspondant], Jonathan Detchart [engineer], Ferdaouss Mattoussi [PhD student].

The goals of the OpenFEC.org http://openfec.org are:

1. to share IPR-free, open, AL-FEC codes,
2. to share high performance, ready-to-use, open, free, C-language, software codecs
3. to share versatile and automated performance evaluation environments.

This project can be useful to users who do not want to know the details of AL-FEC schemes but do need to use one of them in the software they are designing, or by users who want to test new codes or new encoding or decoding techniques, and who do know what they are doing and are looking for, or by users who need to do extensive tests for certain AL-FEC schemes in a given use-case, with a well defined channel model.

5.10. BitHoc

Participants: Chadi Barakat [correspondant], Thierry Turletti.

BitHoc (BitTorrent for wireless ad hoc networks) enables content sharing among spontaneous communities of mobile users using wireless multi-hop connections. It is an open source software developed under the GPLv3 licence. A first version of BitHoc has been made public. We want BitHoc to be the real testbed over which we evaluate our solutions for the support and optimization of file sharing in a mobile wireless environment where the existence of an infrastructure is not needed. The proposed BitHoc architecture includes two principal components: a membership management service and a content sharing service. In its current form it is composed of PDAs and smartphones equipped with WIFI adapters and Windows Mobile 6 operating system.

See also the web page http://planete.inria.fr/bithoc

- Version: 1.2
- Keywords: Tracker-less BitTorent for mobile Ad Hoc networks
- License: GPL (GPLv3)
- Type of human computer interaction: Windows Mobile 6 GUI
- OS/Middelware: Windows Mobile 6
- Required library or software: OpenSSL(http://www.openssl.org/, GPL), C++ Sockets, GPL)
- Programming languages: C++, C#
- Documentation: doxygen

5.11. TICP

Participant: Chadi Barakat [correspondant].

TICP is a TCP-friendly reliable transport protocol to collect information from a large number of network entities. The protocol does not impose any constraint on the nature of the collected information: availability of network entities, statistics on hosts and routers, quality of reception in a multicast session, weather monitoring, etc. TICP ensures two main things: *(i)* the information to collect arrives entirely and correctly to the collector where it is stored and forwarded to upper layers, and *(ii)* the implosion at the collector and the congestion of the network are avoided by controlling the rate of sending probes. The congestion control part of TICP is designed with the main objective to be friendly with applications using TCP. Experimental results show that TICP can achieve better performance than using parallel TCP connections for the data collection. The code of TICP is available upon request, it is an open source software under the GPLv3 licence.

See also the web page http://planete.inria.fr/ticp/

- Version: 1.0
- Keywords: Information Collection, Congestion and Error Control
- License: GPL (GPLv3)
- Type of human computer interaction: XML file
- OS/Middelware: Linux/Unix
- Required library or software: C/C++ Sockets
- Programming languages: C/C++
- Documentation: Text

5.12. Private Data Publication

Participants: Gergely Acs, Claude Castelluccia.

We are developing a set of tools to privately publish different types of datasets. For example, we are developing a software that can be used to sanitize sequential data (described in our CCS paper [41]). The code generates the set of noisy n-grams and generate a synthetic, and private, dataset. We are also developing a tool that implement the histogram sanitization algorithm described in our ICDM paper [33].

These tools are accessible here: http://planete.inrialpes.fr/projects/p-publication/

5.13. Experimentation Software

ACQUA

ACQUA stands for Application for Collaborative Estimation of the Quality of Internet Access. It has been developed within the French National project ANR CMON on Collaborative Monitoring in conjunction with Grenouille.com. ACQUA consists of a tool that lets the user have an estimation of the anomalies of the Internet based on active measurements of end-to-end delay metrics among a predefined set of landmarks (i.e. test points). When an anomaly is detected it is expressed in terms of how many destinations are affected by this anomaly, and how important in terms of delay variation is this anomaly for these affected destinations. See also http://planete.inria.fr/acqua/ for more information and for a java version of the code.

WisMon

WisMon is a Wireless Statistical Monitoring tool that generates real-time statistics from a unified list of packets, which come from possible different probes. This tool fulfills a gap on the wireless experimental field: it provides physical parameters on realtime for evaluation during the experiment, records the data for further processing and builds a single view of the whole wireless communication channel environment. WisMon is available as open source under the Cecill license, at http://planete.inria.fr/software/WisMon/.

WEX Toolbox

The Wireless Experimentation (WEX) Toolbox aims to set up, run and make easier the analysis of wireless experiments. It is a flexible and scalable open-source set of tools that covers all the experimentation steps, from the definition of the experiment scenario to the storage and analysis of results. Sources and binaries of the WEX Toolbox are available under the GPLv2 licence at https://twiki-sop.inria.fr/twiki/bin/view/Projets/Planete/WEXToolkit. WEX Toolbox includes the CrunchXML utility, which aims to make easier the running and the analysis of wireless experimentations. In a nutshell, it implements an efficient synchronization and merging algorithm, which takes XML (or PDML) input trace files generated by multiple probes, and stores only the packets fields that have been marked as relevant by the user in a MySQL database –original pcap traces should be first formatted in XML using wireshark. These operations are done in a smart way to balance the CPU resources between the central server (where the database is created) and the different probes (i.e., PC stations where the capture traces are located). CrunchXML is available under the GNU General Public License v2 at http://twiki-sop.inria.fr/twiki/bin/view/Projets/Planete/CrunchXML.

WiMAX ns-3

This simulation module for the ns-3 network simulator is based on the IEEE 802.16-2004 standard. It implements the PMP topology with TDD mode and aims to provide detailed and standard compliant implementation of the standard, supporting important features including QoS scheduling services, bandwidth management, uplink request/grant scheduling and the OFDM PHY layer. The module is available under the GNU General Public License at http://code.nsnam.org/iamine/ns-3-wimax. It will be included in the official 3.8v release of ns-3.

MonLab

Monitoring Lab is a platform for the emulation and monitoring of traffic in virtual ISP networks. It is supported by the FP7 ECODE project and is available for download at the web page of the tool http://planete.inria.fr/MonLab/ under the terms of the GPL licence. MonLab presents a new approach for the emulation of Internet traffic and for its monitoring across the different routers of the emulated ISP network. In its current version, the traffic is sampled at the packet level in each router of the platform, then monitored at the flow level. We put at the disposal of users real traffic emulation facilities coupled to a set of libraries and tools capable of Cisco NetFlow data export, collection and analysis. Our aim is to enable running and evaluating advanced applications for network wide traffic monitoring and optimization. The development of such applications is out of the scope of this research. We believe that the framework we are proposing can play a significant role in the systematic evaluation and experimentation of these applications' algorithms. Among the direct candidates figure algorithms for traffic engineering and distributed anomaly detection. Furthermore, methods for placing monitors, sampling traffic, coordinating monitors, and inverting sampling traffic will find in our platform a valuable tool for experimentation.

MobiTrade

MobiTrade is the ns-3 and Android implementation of our solution for trading content between wireless devices. The application provides a utility driven trading system for efficient content dissemination on top of a disruption tolerant network. While simple tit-for-tat (TFT) mechanisms can force nodes to *give one to get one*, dealing with the inherent tendency of peers to take much but give back little, they can quickly lead to deadlocks when some (or most) of interesting content must be somehow fetched across the network. To resolve this, MobiTrade proposes a trading mechanism that allows a node (*merchant*) to buy, store, and carry content for other nodes (its *clients*) so that it can later trade it for content it is personally interested in. To exploit this extra degree of freedom, MobiTrade nodes continuously profile the type of content requested and the collaboration level of encountered devices. An appropriate utility function is then used to collect an optimal inventory that maximizes the expected value of stored content for future encounters, matched to the observed mobility patterns, interest patterns, and collaboration levels of encountered nodes. See also http://planete.inria.fr/MobiTrade.

RAP Project-Team (section vide)

SOCRATE Team

5. Software

5.1. WSnet

Socrate is an active contributor to WSnet (http://wsnet.gforge.inria.fr/) a multi-hop wireless network discrete event simulator. WSnet was created in the ARES team and it is now supported by the D-NET team of Inria Rhône-Alpes.

5.2. Wiplan

Wiplan is a software including an Indoor propagation engine and a wireless LAN optimization suite, which has been registered by INSA-Lyon. The heart of this software is the propagation simulation core relying on an original method, MR-FDPF (multi-resolution frequency domain ParFlow). The discrete ParFlow equations are translated in the Fourier domain providing a wide linear system, solved in two steps taking advantage of a multi- resolution approach. The first step computes a cell-based tree structure referred to as the pyramid. In the second phase, a radiating source is simulated, taking advantage of the pre-processed pyramidal structure. Using of a full-space discrete simulator instead of classical ray-tracing techniques is a challenge due to the inherent high computation re- quests. However, we have shown that the use of a multi-resolution approach allows the main computation load to be restricted to a pre-processing phase. Extensive works have been done to make predictions more realistic. The network planning and optimization suite is based on a multi-criteria model relying on a Tabu solver. The development of the wiplan software is a part of the european project iPlan (IAPP-FP7 project). See also the web page http://wiplan.citi.insa-lyon.fr.

TREC Project-Team

5. Software

5.1. Gibbs' Sampler

Participants: François Baccelli, Chung Shue Chen.

The work on the self optimization of cellular networks based on Gibbs' sampler (see Section 6.1.1.5) carried out wit Chung Shue Chen (ALU) in the joint laboratory with Alcatel-Lucent, led to the development of a software prototype that was presented by L. Roullet at the Inria Alcatel-Lucent joint laboratory seminar in November 2012.

5.2. PSI2

Participant: Ana Bušić.

The work on perfect sampling (see Section 6.2.2) has been partially implemented in a software tool PSI2, in collaboration with MESCAL team [Inria Grenoble - Rhône-Alpes]; https://gforge.inria.fr/projects/psi.

URBANET Team

5. Software

5.1. WSNet.

UrbaNet is an active contributor to WSnet (http://wsnet.gforge.inria.fr/), a discrete event simulator dedicated to large scale wireless networks developed and maintained by members of Inria and CITI lab. A major part of this contribution is represented by the implementation of state of the art protocols for medium access control and routing.

The WSNet simulation results obtained following this process are sometimes used as an input for another part of our development effort, which consists in prototype software based on the combination of CPLEX and AMPL for solving mixed integer linear programming problems with column generation.

5.2. TAPASCologne vehicular mobility dataset.

Based on the data made available by the Institute of Transportation Systems at the German Aerospace Center (ITS-DLR), the dataset aims at reproducing, with a high level of realism, car traffic in the greater urban area of the city of Cologne, Germany. To that end, different state-of-art data sources and simulation tools are brought together, so to cover all of the specific aspects required for a proper characterization of vehicular traffic:

- The street layout of the Cologne urban area is obtained from the OpenStreetMap (OSM) database;
- The microscopic mobility of vehicles is simulated with the Simulation of Urban Mobility (SUMO) software;
- The traffic demand information on the macroscopic traffic flows across the Cologne urban area (i.e., the O/D matrix) is derived through the Travel and Activity PAtterns Simulation (TAPAS) methodology;
- The traffic assignment of the vehicular flows described by the TAPASCologne O/D matrix over the road topology is performed by means of Gawron's dynamic user assignment algorithm.

The resulting synthetic trace of the car traffic in a the city of Cologne covers a region of 400 square kilometers for a period of 24 hours, comprising more than 700.000 individual car trips. More information is available on the project website at http://kolntrace.project.citi-lab.fr/ .