RESEARCH CENTER

FIELD
Networks, Systems and Services, Distributed Computing

Activity Report 2014

Section Software

Edition: 2015-03-24

DISTRIBUTED SYSTEMS AND MIDDLEWARE
1. ASAP Project-Team 5
2. ATLANMOD Project-Team 9
3. CIDRE Project-Team 15
4. COAST Team 16
5. CTRL-A Exploratory Action 17
6. MIMOVE Team 18
7. MYRIADS Project-Team 23
8. REGAL Project-Team 27
9. SCALE Team 29
10. SPIRALS Team 31
11. WHISPER Team 34
DISTRIBUTED AND HIGH PERFORMANCE COMPUTING
12. ALGORILLE Project-Team 37
13. ALPINES Project-Team 40
14. AVALON Project-Team 41
15. HIEPACS Project-Team 45
16. KerData Project-Team 50
17. MESCAL Project-Team 53
18. MOAIS Project-Team 56
19. ROMA Team 60
20. RUNTIME Team 61
21. TYREX Project-Team 66
DISTRIBUTED PROGRAMMING AND SOFTWARE ENGINEERING
22. ASCOLA Project-Team 69
23. DIVERSE Project-Team 72
24. FOCUS Project-Team 75
25. INDES Project-Team 78
26. PHOENIX Project-Team 81
27. RMOD Project-Team 89
28. TACOMA Team 91
NETWORKS AND TELECOMMUNICATIONS
29. COATI Project-Team 92
30. DANTE Team 93
31. DIANA Team 94
32. DIONYSOS Project-Team 98
33. DYOGENE Project-Team 99
34. FUN Project-Team 100
35. GANG Project-Team (section vide) 102
36. HIPERCOM2 Team 103
37. INFINE Team 105

38. MADYNES Project-Team 107
39. MAESTRO Project-Team 110
40. MUSE Team 111
41. RAP Project-Team (section vide) 113
42. SOCRATE Project-Team 114
43. URBANET Team 115

ASAP Project-Team

5. New Software and Platforms

5.1. MediEgo: A recommendation solution for webmasters

Participants: Jacques Falcou, Arnaud Jégou, Xavier Lucas, Anne-Marie Kermarrec, Jean-François Verdonck.

Contact:	Anne-Marie Kermarrec
Licence:	Proprietary
Presentation:	Recommendation solution for webmasters
Status:	Beta version, IDDN.FR.001.490030.000.S.P.2013.000.30000 on 09/12/2013

MEDIEGO is a solution for content recommendation based on the users navigation history. The solution 1) collects the usages of the Web users and store them in a profile; 2) uses this profile to associate to each user her most similar users; 3) leverages this implicit network of close users in order to infer their preferences and recommend advertisements and recommendations. MEDIEGO achieves scalability using a sampling method, which provides very good results at a drastically reduced cost. The MEDIEGO recommendation engine is built in collaboration with Sébastien Campion. We have demonstrated the software at the conference Le Web 14 (9-11 Dec 2014) in collaboration with France Television.

5.2. AllYours-P2P: A distributed news recommender (former WhatsUp)

Participants: Heverson Borba Ribeiro, Raziel Carvajal Gomez, Davide Frey, Arnaud Jégou, Anne-Marie Kermarrec.

Contact:	Davide Frey
Licence:	AGPL 3.0
Presentation:	A distributed news recommender
Status:	Beta version, IDDN.FR.001.500002.000.S.P.2013.000.30000 on 09/12/2013

Within the context of the AllYours EIT/ICT-Labs project, we refined the implementation of WhatsUp into the Peer-to-Peer AllYours application. AllYours-P2P is a peer-to-peer based news recommender system that organizes users into an implicit social network based on their explicit opinions. In AllYours-P2P the recommendation process is collaboratively performed by connected users. Every user runs a symmetric piece of software responsible for storing user interests and calculating the affinity between a user and its neighborhood. The local computed similarity then is used to keep virtual connections to other users whose interests are alike and remove connection to the ones that are not. As a result, an interest-based overlay is built and users converge to groups of similar interests within which news are disseminated. The AllYours-P2P software consists of two parts, running on each peer: an embedded application server, based on Jetty, and a web interface accessible from any web browser. The back-end is written in Java, while the user interface comprises HTML and Javascript code. AllYours-P2P is currently available in three different platforms: Mac OSx (10.5 or later), Windows (Vista and Windows 7) and Linux (Ubuntu 10.4 or later). We have tested Allyours-p2p in a real life environment with a set of invited users in Italy from Sep to Nov 2014. These test were a part of joint project between ASAP Team and its Italian partner Trentorise.

Currently the implementation of Allyours-p2p includes approximately 21K lines of code.

5.3. HyRec: A hybrid recommender system

Participants: Davide Frey, Anne-Marie Kermarrec.

Contact:	Davide Frey
Licence:	Proprietary
Status:	Beta version, IDDN.FR.001.500007.000.S.P.2013.000.30000 on 09/12/2013

This work leads to the development of HyRec, a hybrid recommender system. The motivation of this work is to explore solutions that could in some sense democratize personalization by making it accessible to any content provider company without generating huge investments. HyRec implements a user-based collaborative filtering scheme and offloads CPU-intensive recommendation tasks to front-end client browsers, while retaining storage and orchestration tasks within back-end servers. HyRec seeks to provide the scalability of p2p approaches without forcing content providers to give up the control of the system. This software has been developed in collaboration with Antoine Boutet (Univ. Saint-Étienne) and Rhicheek Patra (EPFL).

5.4. GossipLib: A library for gossip-based applications

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

Contact:	Heverson Borba Ribeiro, Davide Frey
Licence:	AGPL 3.0
Presentation:	Library for gossip protocols
Status:	Alpha version, IDDN.FR.001.500001.000.S.P.2013.000.10000 on 09/12/2013

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 application, and provides also 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. Currently the implementation of GossipLib includes approximately 9K lines of code, and is used in several projects by ASAP, including HEAP, AllYours-P2P, and Behave.

5.5. YALPS: A library for p2p applications

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

Contact:	Heverson Borba Ribeiro, Davide Frey
Licence:	Open Source
Presentation:	Library for p2p applications
Status:	Beta version, IDDN.FR.001.500003.000.S.P.2013.000.10000 on 09/12/2013

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. 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 implementation of YALPS includes approximately 16K lines of code, and is used in several projects by ASAP, including HEAP, AllYours-P2P, and Behave. This work was done in collaboration with Maxime Monod (EPFL).

5.6. HEAP: Heterogeneity-aware gossip protocol

Participants: Davide Frey, Arnaud Jégou, 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.

5.7. Brow2Brow: Browser-to-browser serverless toolboxes

Participants: Raziel Carvajal Gomez, Davide Frey, Anne-Marie Kermarrec.

Brow2Brow is an "Action de Development Technologique", i.e. a collaborative development project that aims at providing a middleware and software library for browser-to-browser applications. Brow2Brow involves the ASAP team as well as the DICE Team from Inria Grenoble (Antenne de Lyon). The project seeks to provide an alternative to the current model followed by Web2.0 applications by exploiting the recently introduced WebRTC standard. Existing Web 2.0 applications collect data on browsers and send it to servers that store and process it. The goal of Brow2Brow is to provide an alternative approach where browsers can themselves proceed to collaborative data processing. This will make it possible avoid data concentration at a single server. The project has resulted so far in the development of WebGC, a library for gossip-based applications on browsers.

5.8. WebGC: Web-based Gossip Communication

Participants: Raziel Carvajal Gomez, Davide Frey, Anne-Marie Kermarrec.

Contact:	Raziel Carvajal Gomez, Davide Frey
License:	Not-yet released
Presentation:	Library for Gossip protocols within Web Browsers
Status:	Ongoing development

WebGC is a library for gossip-based communication between web-browsers. It has been developed in collaboration with Mathieu Simonin in the context of the Brow2Brow ADT project. WebGC builds on the recent WebRTC standard as well as on PeerJS, an open-source project that provides primitives for data transfer on top of WebRTC.

The library currently includes the implementation of two peer sampling protocols, CYCLON and the generic peer-sampling protocol from [7], as well as a clustering protocol [1]. All protocols implement a common GOSSIPPROTOCOL "interface"—since Javascript does not natively support interfaces, we adopt the interface pattern. A COORDINATOR makes it possible to stack these protocols on top of each other to implement applications.

ATLANMOD Project-Team

5. New Software and Platforms

5.1. The ATL Model Transformation Language

URL: http://www.eclipse.org/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 [48]. 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.

Performing just the required work on model transformation improves scalability, an open issue the previous described works contribute to solve.n efficient execution, as in the the lazy and reactive scenarios, may help with scalability problems by focusing the tasks in the required part of a very large transformation. However, this is not always the case and we might have to perform operations in the whole model. In this scenario, a solution for the scalability problem would be to take advantage of multi-core architectures that are very popular today, to improve computation times in the transformation of very large models. In this sense, a first step explores the strong parallelization properties rule-based languages like ATL have. A new prototype implementation of a parallel ATL engine has been developed showing how transformations can be developed without taking into account concurrency concerns, and such a transformation engine can automatically parallelize operations improving execution times.

Aligned with this research line we propose in recent works an approach to automatically parallelize the computation of model transformation using Cloud infrastructures. For this, we take advantage of a well-known distributed programming model: *MapReduce*. In this sense, we introduce an algorithm aligning both execution semantics of ATL and MapReduce. Based on this, a new prototype tool has been developed [0] showing in several experiments the scalability of the solution.

5.2. MoDisco (Model Discovery)

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

[0] https://github.com/atlanmod/ATL_MR

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 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 [0], 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://atlanmod.github.io/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 design it and make it evolve. Since the very first implementation of the Collaboro toolset was released, it has evolved to provide support to both Eclipse-based and web-based clients.

The Eclipse-based client has been developed as a plugin in the platform while the web-based client includes two components: (1) the server-side part, which offers a set of services to access to the main functionalities of Collaboro; and the client-side part, which allows both end-users and developers to take part of the DSML development process from their browsers. The server-side component has been developed as a Java web application which uses a set of Servlets providing the required services. On the other hand, the client-side component has been developed as an AngularJS-enabled website and provides.

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

The Collaboro clients provide access to the following features:

- Version view to navigate through the Proposals of a version of a language. 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 menu bar.
- Notation engine and Notation view to render SVG snapshots of the DSL concrete syntax.
- Support for example-driven development of DSMLs, thus incoporating a graphical editor which allows end-users to draw examples of the DSML they are developing.

5.4. JSON Discoverer

URL: http://atlanmod.github.io/json-discoverer/

Given a set of JSON documents, the tool (distributed as an open source Eclipse plugin contributed to MoDisco) returns a model describing their implicit schema. We follow an iterative process where new JSON documents (from the same or different services within the API) contribute to enrich the generated model. The model helps to both understand single services and to infer possible relationships between them, thus suggesting possible compositions and providing an overall view of the application domain. The tool has also been released as a web site, thus allowing any web developer to use our approach without the need of installing Eclipse.

5.5. EMF-REST

URL: http://emf-rest.com/

EMF is *the modeling framework* of the Eclipse community. While EMF is able to automatically generate Java APIs from Ecore models, it is still missing support to deal with Web APIs such as RESTful ones that could boost the use of modeling techniques in the Web. However, the creation of RESTful APIs requires from developers not only an investment in implementation but also a good understanding of the REST Principles to apply them correctly. We therefore created EMF-REST, a tool that empowers EMF to get Truly RESTFul APIs from Ecore models, thus allowing web developers to generate JSON-based Web APIs for their applications. It generates both a JavaScript API to work with models as Javascript Objects in the client-side (without any EMF dependency) and REST services in the server-side based on the Java JAX-RS specification.

5.6. EMF Views (Model Views)

URL: https://github.com/atlanmod/emfviews

The Eclipse Modeling Framework (EMF) is widely used in the Eclipse community: defining domain models and generating corresponding source code, modeling software architectures, specifying DSL concepts or simply representing software/user data in different contexts. This implies that any software project involves a large number of heterogeneous but interrelated EMF models. To make matters worse, not all participants in the project should have the same kind of access/views on the models. Some users only need to see some parts of one model, others have to get the full model extended with data from another model, or simply access to a combination of information coming from different interconnected models. Up to now, creating such perspectives transparently in EMF was almost impossible. Based on the unquestionable success/usefulness of database views to solve similar problems in databases, EMF Views aims to bring the same concept to the modeling world. Thanks to the three main constructs (inspired from SQL) offered by the tool, designers can create new model views: SELECTing a subset of elements from a model, PROJECTing only some of the properties of those elements and/or JOINing them with elements from other models. A model view is a special type of model whose instances are directly computed at runtime based on the model view definition

and concerned actual model(s). EMF Views has been initially developed in the context of the TEAP industrial project http://www.teap-project.org/ that ended in November 2014, by showing different possible applications of model views including:

- Software architect/developer views relating UML design models and Java code models (cf. Eclipse MoDisco).
- Enterprise architect views linking (BPMN) business process models, (ReqIF) requirements models and (TOGAF) architecture models.
- View transformation using dedicated technologies (e.g. Eclipse ATL).
- Report generation from views, etc.

The EMF Views prototype is currently being re-used and further developed, in a (meta)model extension context this time, within the ongoing MoNoGe industrial project. The objective of this present work is to propose a simple base generic (meta)model extension mechanism, relying on EMF Views capabilities, that could be deployed in different scenarios where (meta)model extension is required (e.g. metamodel evolution, model integration, etc.)

A presentation of EMF Views tooks place at EclipseCon 2014 [0], held in San Francisco, California, U.S.A

5.7. 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 [0]) 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.

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 [0]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 [45] 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.

[0] https://www.eclipsecon.org/na2014/session/modeling-symposium
[0] http://www.eclipse.org/modeling/mdt/?project=ocl
[0] http://eclipseclp.org/

5.8. NeoEMF

URL: http://www.neo4emf.com

NeoEMF (a relaunch of the tool formerly known as Neo4EMF) is an open source software distributed under the terms of the Eclipse Public License that provides a backend-agnostic persistence solution for big, complex and highly interconnected EMF models. NeoEMF is a model repository and persistence framework allowing on-demand loading, storage, and unloading of large-scale EMF models.

NeoEMF is designed to allow the easy integration of custom backends depending on user needs. By default, NeoEMF is bundled with out-of-the-box support for graph databases (based on the blueprints API [0] and key-value stores (based on MapDB [0]). Blueprints is an abstraction layer for graph storages that allows changing the actual database used without affecting the application code. The blueprints-based back-end allows the integration of NeoEMF and Neo4j—among other databases—providing in NeoEMF the full set of features already implemented in Neo4EMF. MapDB is an efficient key-value store that provides concurrent Maps, Sets and Queues backed by disk storage or off-heap memory.

In terms of performance, NeoEMF eases data access and storage not only in a manner to reduce time and memory usage but also to allow big models to fit into small memory. This is provided through the following features:

- Lazy-loading mechanism. Model objects are loaded on demand while needed. In its basic configuration, model objects act as a proxy that ocuppy little memory, and fields are only retrieved when accessed.
- Caching. NeoEMF relies on database caches to retrieve EObjects, but in some situation this is not enough. For this reason, the architecture on NeoEMF allows the easy implementation of domain-specific cache strategies based on the decorator pattern.
- Auto-commit. In back-ends in which transaction data is stored on the heap, it is possible to use the auto-commit feature to split large transaction into several small ones.
- Dirty saving. The dirty saving feature is an step forward on the auto-commit strategy. It allows to safely handle big transactions by spliting them into small ones by saving partial changes made on models to disk. In case of transaction failure or cancellation, the partial model changes can be reverted and the model is restored to its original state.

A session about NeoEMF took place at eclipseCon France 2014 [0], held in Toulouse, France.

Works are still going over NeoEMF (within the context of the project ITM Factory - FUI14) to provide more utilities such as backend-aware query languages (which allows improving performance by taking advantage of the backend built-in query languages), concurrent access, model distribution, and other Ecore utilities.

5.9. GitHub Label Analyzer

URL: http://atlanmod.github.io/gila/

Reporting bugs, asking for new features and in general giving any kind of feedback is the easiest way to contribute to an Open-Source Software (OSS) project. In GitHub, the largest code hosting service for OSS, this feedback is typically expressed as new issues for the project managed by an issue-tracking system available in each new project repository. Among other features, the issue tracker allows creating and assigning labels to issues with the goal of helping the project community to better classify and manage those issues (e.g., facilitating the identification of issues for top priority components or candidate developers that could solve them). Nevertheless, as the project grows a manual browsing of the project issues is no longer feasible.

[0] https://github.com/tinkerpop/blueprints/
[0] https://github.com/jankotek/MapDB
[0] https://www.eclipsecon.org/france2014/session/neo4emf-when-big-models-are-no-longer-issue

We believe that visualization techniques could be applied here to overcome this challenge. In particular, we have created GiLA, a tool to better understand how labels are being used in GitHub projects, with the aim of providing more insights into how such projects are being managed. GiLA provides three visualizations addressing three different viewpoints, specifically:

V1 Label usage, which helps to identify the most used labels and which ones are commonly used together.

V2 User involvement, which allows discovering the most active and knowledgeable users around each label.

V3 Typical Label timeline, which provides some insights about how issues under that label evolve over time (e.g., time to be treated).

The tool can be used to explore these viewpoints on all the original projects (i.e., projects that are not a fork of a previous project) in GitHub. We believe that the results favour not only a better comprehension of the project but also help in its advancement, e.g., by helping to quickly identify experts on a particular topic/label.

CIDRE Project-Team

5. New Software and Platforms

5.1. Intrusion Detection and Privacy

Members of the team have developed several intrusion detectors and security tools: **Blare** implements our approach of illegal information flow detection at the OS level for a single node and a set of nodes; **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; **Netzob** is an open-source tool for reverse engineering, traffic generation and fuzzing of communication protocols; a log visualization tool called **ELVIS** (Extensible Log VISualization) has been implemented in order to test our approches for log exploration.

In addition, the team participate to the development of **GEPETO** (GEoPrivacy-Enhancing TOolkit), an open source software for managing location data (in cooperation with the CNRS Lab. LAAS, Toulouse). GEPETO can be used to visualize, sanitize, perform inference attacks, and measure the utility of a particular geolocated dataset.

These tools are still under development in the team. Nevertheless, there are not new. For more details, please see previous activity reports.

COAST Team

4. New Software and Platforms

4.1. Rivage

Participant: Claudia-Lavinia Ignat [contact].

Rivage (Real-tIme Vector grAphic Group Editor) is a real-time collaborative graphical editor. Several users can edit at the same time and in real-time a graphical document, user changes being immediately seen by the other users. The editor relies on a peer-to-peer architecture where users can join and leave the group at any time. Each user has a copy of the shared document and user changes on the document copies are merged in real-time by using a CRDT (Commutative Replicated Data Type) algorithm. The code is available at https://github.com/stephanemartin/rivage/

4.2. Replication Benchmarker

Participants: Pascal Urso [contact], Mehdi Ahmed-Nacer, Gérald Oster.

The Replication Benchmarker is a performance evaluation framework for optimistic replication mechanisms used in collaborative applications. It contains a library of implementation of several CRDT (Commutative Replicated Data Type) and OT (Operational Transformation) algorithms for different data types: text, set, trees. The framework is able to evaluate the performance of comparable algorithms on different corpus of events traces. These events traces can be produced randomly according to different parameters, can be extracted from real real-time editing session that have been recorded, or can be automatically extracted from distributed version control repositories such as the one produced with Git. Performances of the algorithms are measured in term of execution time, memory footprint and merge result quality (compared to manual merge history stored in git repositories). The source code of this evaluation framework is available at https://github.com/score-team/replication-benchmarker/.

4.3. BeGoood

Participant: Gérôme Canals.

BeGoood is a generic system for managing non-regression tests on knowledge bases. BeGoood allows to define test plans in order to monitor the evolution of knowledge-bases. Any system answering queries by providing results in the form of set of strings can be tested with BeGoood. BeGoood has been developed following a REST architecture and is independent of any application domain. BeGoood is a part of the Kolflow infrastructure and is available at https://github.com/kolflow/.

4.4. MUTE

Participants: Claudia Ignat, Luc André, François Charoy, Gérald Oster [contact].

MUTE (Multi-User Text Editor) is a web-based text editing tool that allows to edit documents collaboratively in real-time. It implements our recent work on collaborative editing algorithms and more specifically the LOGOOTSPLIT+ approach [22]. Compared to existing web-based collaborative text editing tool this editor does not require a powerful central server since the server is not performing any computation and acts as a simple broadcast server. Our editor offers support for working offline while still being able to reconnect at a later time. This prototype is distributed under the term of GNU GPLv3 licence and is freely available at https://github.com/score-team/mute-demo/. A demo server is hosted at http://mute-editorcrdt.rhcloud.com/.

CTRL-A Exploratory Action

5. New Software and Platforms

5.1. Heptagon/BZR programming language

Participants: Gwenaël Delaval [Contact person], Eric Rutten.

We want to produce results concretely usable by third parties, either in cooperative projects, or by free diffusion of tools. One perspective is to build tool boxes for the design of continuous control solutions for computing systems: it will be explored in the future. A readily available result concerns discrete control and programming.

HEPTAGON is a dataflow synchronous language, inspired from LUCID SYNCHRONE[0]. Its compiler is meant to be simple and modular, allowing this language to be a good support for the prototyping of compilation methods of synchronous languages.

HEPTAGON has been used to build BZR [0], which is an extension of the former with contracts constructs. These contracts allow to express dynamic temporal properties on the inputs and outputs of HEPTAGON node. These properties are then enforced, within the compilation of a BZR program, by discrete controller synthesis, using the SIGALI tool [0]. The synthesized controller is itself generated in HEPTAGON, allowing its analysis and compilation towards different target languages (C, Java, VHDL).

Prospects about Heptagon/BZR lie in: support for programming methodologies in BZR: debug, diagnosis, abstraction and composition, ...; extensions towards use of more expressive synthesis tools ; integration of target code into execution platforms (Fractal, reconfigurable FPGA, ...)

[0] http://www.di.ens.fr/~pouzet/lucid-synchrone
[0] http://bzr.inria.fr
[0] http://www.irisa.fr/vertecs/Logiciels/sigali.html

MIMOVE Team

5. New Software and Platforms

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 [4]. The focus of our 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 (NSs) that have compatible functionalities to interact despite mismatching interfaces and/or behaviors.

The Discovery Enabler is the component of the overall CONNECT architecture that handles discovery of networked systems, 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 Synthesis Enabler assumes semantically-annotated system descriptions *à la* OWL-S, which are made available by the Discovery Enabler, together with a domain ontology, and produces the mediators that enable functionally compatible networked systems to interoperate. The semantically-annotated interfaces of the NSs that need to communicate are processed to compute the semantic mapping between their respective operations using a constraint solver. The resulting mapping serves generating a mediator that coordinates the behaviors of the NSs and guarantees their successful interaction. Only when the mediator includes all the details about the communication of NSs, can interoperability be achieved, which calls for the adequate concretization of synthesized mediators.

The *concretization of mediators* bridges the gap between the application level, which provides the abstraction necessary to reason about interoperability and synthesize mediators, and the middleware-level, which provides the techniques necessary to implement these mediators. Concretization entails the instantiation of the data structures expected by each NS and their delivery according to the interaction pattern defined by the middleware, based on which the NS is implemented. Therefore, we have been developing a mediation engine that, besides executing the data translations specified by the mediator, generates composed parsers and composers, which can process complex messages, by relying on existing libraries associated with standard protocols and state-of-the-art middleware solutions.

The data structures defined within OWL-S system descriptions, that is, the types of the inputs and outputs of the OWL-S atomic process were previously defined manually. As part of the prototype implementation that allows the mediator engine to generate composed parsers and composers, we made an extension that enables the inference of correct data-types [17], relieving developers from the time-consuming task of defining them. This prototype takes the form of a JAVA library that can optionally be used by the Synthesis Enabler whenever abstract data structures are unavailable. This library can be obtained from the iCONNECT GIT repository https://gforge.inria.fr/git/iconnect/iconnect.git (under the subproject `mtc`). While the underlying source code closely follows the formal mechanisms (such as tree automata) and algorithms presented in [17], we further concerned ourselves with making this library usable for non-expert developers by adhering to well-established standards. Specifically, data types, which are internally modeled as top-down tree automata, are transformed both on the input and output to RelaxNG (http://relaxng.org/) or XSD (www.w3.org/TR/xmlschema-1/).

The iCONNECT software has been released in the OW2 open source community (http://forge.ow2.org/projects/iconnect/), as part of FISSi, the Future Internet Software and Services initiative (http://www.ow2.org/view/Future_Internet/). OW2 and FISSi will give to this effort the required visibility in order to attract users and developers of the open source community.

5.3. XSB: eXtensible Service Bus for the Future Internet

Participant: Nikolaos Georgantas [correspondent].

The eXtensible Service Bus (XSB) is a development and runtime environment dedicated to complex distributed applications of the Future Internet. Such applications will be based, to a large extent, on the open integration of extremely heterogeneous systems, such as lightweight embedded systems (e.g., sensors, actuators and networks of them), mobile systems (e.g., smartphone applications), and resource-rich IT systems (e.g., systems hosted on enterprise servers and Cloud infrastructures). Such heterogeneous systems are supported by enabling middleware platforms, particularly for their interaction. With regard to middleware-supported interaction, the client-service (CS), publish-subscribe (PS), and tuple space (TS) paradigms are among the most widely employed ones, with numerous related middleware platforms, such as: Web Services, Java RMI for CS; JMS, SIENA for PS; and JavaSpaces, Lime for TS. XSB then provides support for the seamless integration of heterogeneous interaction paradigms (CS, PS and TS).

In a nutshell, our systematic interoperability approach implemented by the proposed XSB is carried out in two stages. First, a middleware platform is abstracted under a corresponding interaction paradigm among the three base ones, i.e., CS, PS and TS. To this aim, we have elicited a connector model for each paradigm, which comprehensively covers its essential semantics. Then, these three models are abstracted further into a single generic application (GA) connector model, which encompasses their common interaction semantics. Based on GA, we build abstract connector converters that enable interconnecting the base interaction paradigms.

Following the above, XSB is an abstract service bus that prescribes only the high-level semantics of the common bus protocol, which is the GA semantics. Furthermore, we provide an implementation of the XSB, building upon existing SOA and ESB realizations. XSB features richer interaction semantics than common ESBs to deal effectively with the increased Future Internet heterogeneity. Moreover, from its very conception, XSB incorporates special consideration for the cross-integration of heterogeneous interaction paradigms. Services relying on different interaction paradigms can be plugged into XSB by employing binding components (BCs) that adapt between their native middleware and the common bus protocol. This adaptation is based on the abstractions, and in particular on the conversion between the native middleware, the corresponding CS/PS/TS abstraction, and the GA abstraction.

Furthermore, we provide a companion implementation, named Light Service Bus (LSB), targeting the Internet of Things (IoT) domain. LSB forms a concrete access solution for IoT systems as it is able to cope with the diversity of the involved interaction protocols and take care of the specifics of IoT services, such as resource constraints, dynamic environments, data orientation, etc. It is implemented to be lightweight in nature and uses REST as the common protocol/bus in place of an ESB solution. In LSB, we confirm the wide use of the aforementioned interaction paradigms (CS/PS/TS) but also underline the existence of an additional paradigm focused on continuous interaction known as Streaming (STR).

Both the XSB and LSB solutions are available for download under open source licenses at http://xsb.inria.fr and http://websvn.ow2.org/listing.php?repname=choreos&path=%2Ftrunk%2Fextensible-service-access%2Flsb%2FLsbBindingComponents%2F respectively.

5.4. MobIoT: Service-oriented Middleware for the Mobile IoT

Participant: Valérie Issarny [correspondent].

MobIoT is a service-oriented middleware aimed at the mobile Internet of Things (IoT), which in particular deals with the ultra-large scale, heterogeneity and dynamics of the target networking environment. MobIoT offers novel probabilistic service discovery and composition approaches, and wraps legacy access protocols to be seamlessly executed by the middleware. The middleware exposes two levels of service abstractions: Thing as a service (on the service provider side); and Things measurements/actions as a service (on the service consumer side).

Key features of MobIoT lie in: (i) the exploitation of ontologies to overcome the heterogeneity of the Things network, (ii) the introduction of probabilistic approaches for both registering and retrieving networked things so as to filter out the ones that are redundant with already known alternatives, and finally, (iii) the exploitation of Thing services composition for responding to user queries asking information about the physical world so as to ease interaction with such a complex and dynamic networking environment.

MobIoT is implemented using Java and the Android platform, and consists of two complementary components: The MobIoT Mobile middleware and the MobIoT Web Service. The MobIoT Mobile middleware is deployed on mobile devices (e.g., smartphones, tablets, sensor devices). It wraps: (i) the Query component that enables the querying of the physical world, (ii) the Registration component that deals with the probabilistic registration of local sensors and actuators, (iii) the domain ontology that allows reasoning about the features of Things, and (iv) the Sensor Access component that enables the sensor data retrieval and exposure. The MobIoT Web Service wraps: (i) the Registry component that keeps tracks of the registered services, (ii) the probabilistic Lookup component that allows retrieving relevant services in a scalable way, and (iii) the Composition & Estimation component to answer queries over the physical world using available Thing services, and finally domain and devices ontologies.

The MobIoT middleware is available for download under an open source license at http://mobiot.ow2.org.

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 the 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. Dioptase and Spinel: Lightweight Streaming Middleware for the IoT

Participant: Valérie Issarny [correspondent].

Dioptase is a service-oriented middleware for developing stream-based applications which produce, process, store and consume data streams in complex environments such as the Internet of Things (IoT) and Wireless Sensor and Actuator Networks (WSAN). Dioptase leverages a novel service-oriented architecture for continuous processing, in order to deal with the large scale and the heterogeneity of the IoT. Once Dioptase is deployed onto a device, it enables developers to manage it as a generic pool of resources that can execute tasks provided over time. Those tasks are described as compositions of both standard continuous processing operators and customized computations written using a new lightweight stream processing language, called *DiSPL*.

The IoT infrastructure is composed of various devices (sensors, registries, proxies, clusters, etc.), and Dioptase is intended to be deployed onto all of them. To this end, Dioptase is highly modular and can be customized depending on the targeted devices and their roles in the IoT:

- *Dioptase core* is the base version of Dioptase that enables developers to (i) manage embedded sensors and actuators of a device through services and (ii) deploy tasks onto the device at any time.
- *Dioptase task mapper* is a server that implements our research on task mapping and automated deployment. Given a task graph, this server computes where to deploy each task according to the characteristics of tasks and available devices, and then manages the execution of the deployed tasks over time.
- *Dioptase proxy* is a pub/sub broker that enables interactions between Things that can not communicate directly, because of non-compatible networking interfaces/protocols or the use of address translation techniques.
- *Dioptase exchange* is a privacy proxy that manages the data and the services provided by a network of Things. It authenticates outsider Things and users, enabling them to request data streams or services according to access control and data-accuracy policies.

As part of the design of Dioptase, we have been investigating how to open legacy sensors to the future IoT. Toward this end, we propose to take advantage of the multi-modal connectivity as well as the mobility of smartphones, using them as mobile proxies that opportunistically discover close-by static sensors and act as intermediaries between them and the IoT. Spinel is a prototype of such an opportunistic proxy for mobile phones, which monitors the smartphone's mobility and further infers when to discover and register the sensors to an IoT discovery infrastructure, for instance the MobIoT registry(§ 5.4). Spinel collects data from the close-by sensors and pushes the collected data to an IoT stream processing infrastructure, for instance the Dioptase middleware. We will shortly release both Dioptase and Spinel under open source licenses.

5.7. 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, which can be used to develop mobile social applications.

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/.

MYRIADS Project-Team

4. New Software and Platforms

4.1. ConPaaS

Contact: Guillaume Pierre, Guillaume.Pierre@irisa.fr

URL: http://www.conpaas.eu/

Status: Version 1.4.2

License: BSD

Presentation: ConPaaS [60] is a runtime environment for hosting applications in the cloud. It aims at offering the full power of the cloud to application developers while shielding them from the associated complexity of the cloud. ConPaaS is designed to host both high-performance scientific applications and online Web applications. It automates the entire life-cycle of an application, including collaborative development, deployment, performance monitoring, and automatic scaling. This allows developers to focus their attention on application-specific concerns rather than on cloud-specific details.

Active contributors (from the Myriads team): Eliya Buyukkaya, Ancuta Iordache, Morteza Neishaboori, Guillaume Pierre, Dzenan Softic, Genc Tato, Teodor Crivat.

Impact: ConPaaS is recognized as one of the major open-source PaaS environments. It is being developed by teams in Rennes, Amsterdan, Berlin and Ljubljana. Technology transfer of ConPaaS technology is ongoing in the context of the MC-DATA EIT ICT Labs project.

4.2. HOCL-tools

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

Status: Version 1.0 to be released in open source

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 and Myriads teams. 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 higher 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, we recently extended the HOCL compiler so as to support 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): Matthieu Simonin, Cédric Tedeschi, Javier Rojas Balderrama.

Impact: The compiler is used as a tool within the team to develop HOCL programs. The decentralized workflow execution support has been extensively used to produce results published and presented at several conferences. It is also used in the framework of the DALHIS [0] associated team, as a workflow template executor, integrated with the TIGRES workflow manager developed at the Lawrence Berkeley National Lab. It is supported by the GinFlow ADT funded by Inria.

4.3. Merkat

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

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

Status: Version 1.0

License: TBD

Presentation: Merkat is a market-based private PaaS (Platform-as-a-Service) system, supporting dynamic, fine-grained resource allocation and automatic application management [49], [48] [3]. Merkat implements a proportional-share auction that ensures maximum resource utilization while providing incentives to applications to regulate their resource usage. Merkat 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. Merkat is implemented in Python and uses OpenNebula for virtual machine management. Experimental results on the Grid'5000 testbed show that using Merkat increases resource utilization and improves application performance. Merkat is currently being evaluated by EDF R&D using EDF high-performance applications. The development was initiated in the framework of Stefania Costache PhD's thesis.

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

Impact: Merkat has been integrated in EDF R&D portal providing access to internal computing resources and is currently used on a testbed at EDF R&D.

4.4. Meryn

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

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

Status: Version 1.0

License: TBD

Presentation: Meryn is an open, SLA-driven PaaS architecture that supports cloud bursting and allows hosting an extensible set of application types. Meryn relies on a decentralized optimization policy that aims at maximizing the overall provider profit, taking into account the penalties incurred when quality guarantees are unsatisfied [51]. The current Meryn prototype was implemented using shell scripts, builds upon the Snooze VM manager software, and supports batch and MapReduce applications using respectively the Oracle Grid Engine OGE 6.2u7 and Hadoop 0.20.2 frameworks. Meryn was developed in the framework of Djawida Dib's PhD thesis [10].

Active contributors (from the Myriads team): Djawida Dib, Christine Morin, Nikolaos Parlavantzas.

Impact: Meryn is not yet distributed as open source.

4.5. Resilin

[0] http://project.inria.fr/dalhis

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

URL: http://resilin.inria.fr

Status: Version 1.0

License: GNU Affero GPL

Presentation: Resilin [6] 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, Céline Merlet, Christine Morin, Nikolaos Parlavantzas, Matthieu Simonin.

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

4.6. Snooze

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

URL: http://snooze.inria.fr

Status: Version 2.1.5

License: GPLv2

Presentation: Snooze [53], [52], [54] [4] is 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 [59]), and power management to transition idle servers into a power saving mode. Snooze is a highly modular piece of 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. Cassandra (since 2.0.0) can be used as base backend, providing reliability and scalability to the database management system. At a higher level Snooze provides its own REST API as well as an EC2 compatible API (since 2.1.0). It can thus be controlled from the command line (using the legacy client or an EC2 compatible tool), or from different langage libraries (libcloud, jcloud ...). Snooze also provides a web interface to control the system.

Snooze was used as a building box for two internships projects during the summer of 2014. The EC2 interface was used to execute Hadoop jobs configured by Resilin software. As a result we show that (1) the EC2 interface was expressive enough to work with a higher level tool and (2) the control over Snooze allow a better placement of data chunks for Hadoop jobs which leads to a better reliability

of the execution of the different jobs. The second internship topic took part in a collaboration with the Northeastern University of Boston. The goal was to build a *Checkpoint as a Service* system. The service allows users to execute their computations in a cloud environment in a reliable way. Periodic checkpoints are saved making it possible to restore the computation from a previous state in case of failures. This work is described in [31].

Active contributors (from Myriads team): Jiajun Cao, Gene Cooperman, Eugen Feller, Yvon Jégou, David Margery, Christine Morin, Matthieu Simonin.

Impact: Snooze has been used by students at LIFL, IRIT in France and LBNL in the US in the framework of internships. It has also been deployed and experimented at EDF R&D. Snooze entry won the 2nd prize of the scalability challenge at CCGrid2013. Finally, we know that it was experimented by external users from academia and industry as we received feed-back from them. Snooze development was supported by the Snooze ADT funded by Inria from October 2012 to September 2014.

4.7. Virtual Execution Platform (VEP)

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

URL: http://project.inria.fr/vep/

Status: Version 2.2

License: BSD

Presentation: Virtual Execution Platform

(VEP) [57] is a Contrail (http://contrail-project.eu) service that sits just above IaaS layer at the service provider end of the Contrail cloud federation. The VEP service 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 service 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 OpenStack IaaS frameworks were initially supported. During the VEP-S EIT ICT Labs activity in 2014, VEP was extended with a new OCCI IaaS driver which allows to control any IaaS framework providing a standard OCCI API. Support for the new OCCI SLA proposition from OGF has also been added and allows to represent the VEP CEEs in a standard format. Finally, during this activity, the Zabbix open source distributed monitoring system was integrated to VEP.

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

Impact: VEP is part of Contrail software stack. External users can experiment with it using the open testbed operated by Myriads team. Technology transfer of VEP technology is ongoing in the context of the VEP-S EIT ICT Labs activity.

REGAL Project-Team

4. New Software and Platforms

4.1. NumaGiC

Participants: Lokesh Gidra, Marc Shapiro, Julien Sopena [correspondent], Gaël Thomas.

NumaGiC is a version of the HotSpot garbage collector (GC) adapted to many-core computers with very large main memories. In order to maximise GC throughput, it manages the trade-off between memory locality (local scans) and parallelism (work stealing) in a self-balancing manner. Furthemore, the collector features several memory placement heuristics that improve locality. NumaGiC is described in a paper accepted for publication at ASPLOS 2015 [29].

4.2. G-DUR

Participants: Masoud Saeida Ardekani, Dastagiri Reddy Malikireddy, Marc Shapiro [correspondent].

A large family of distributed transactional protocols have a common structure, called Deferred Update Replication (DUR). DUR provides dependability by replicating data, and performance by not re-executing transactions but only applying their updates. Protocols of the DUR family differ only in behaviors of few generic functions. Based on this insight, we offer a generic DUR middleware, called G-DUR, along with a library of finely-optimized plug-in implementations of the required behaviors. This paper presents the middleware, the plugins, and an extensive experimental evaluation in a geo-replicated environment. Our empirical study shows that:

1. G-DUR allows developers to implement various transactional protocols under 600 lines of code;
2. It provides a fair, apples-to-apples comparison between transactional protocols;
3. By replacing plugs-ins, developers can use G-DUR to understand bottlenecks in their protocols;
4. This in turn enables the improvement of existing protocols; and
5. Given a protocol, G-DUR helps evaluate the cost of ensuring various degrees of dependability.

G-DUR and the results of the comparison campaign are described in a paper to Middleware 2014 [33]. This research is supported in part by ConcoRDanT ANR project (Section 7.1.7) and by the FP7 grant SyncFree (Section 7.2.1.1).

Jessy is freely available on github under http://Github.com/msaeida/jessy under an Apache license.

4.3. SwiftCloud

Participants: Mahsa Najafzadeh, Marc Shapiro [correspondent], Serdar Tasiran, Marek Zawirski.

Client-side (e.g., mobile or in-browser) apps need local access to shared cloud data, but current technologies either do not provide fault-tolerant consistency guarantees, or do not scale to high numbers of unreliable and resource-poor clients, or both. Addressing this issue, the SwiftCloud distributed object database supports high numbers of client-side partial replicas. SwiftCloud offers fast reads and writes from a causally-consistent client-side cache. It is scalable, thanks to small and bounded metadata, and available, tolerating faults and intermittent connectivity by switching between data centres. The price to pay is a modest amount of staleness. A recent Inria Research Report (submitted for publication) presents the SwiftCloud algorithms, design, and experimental evaluation, which shows that client-side apps enjoy the same guarantees as a cloud data store, at a small cost.

SwiftCloud is supported by the ConcoRDanT ANR project (Section 7.1.7), by a Google Research Award, and by the FP7 grant SyncFree (Section 7.2.1.1).

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

4.4. Antidote

Participants: Tyler Crain, Marc Shapiro [correspondent], Serdar Tasiran, Alejandro Tomsic.

Antidote is the flexible cloud database platform currently under development in the SyncFree European project (Section 7.2.1.1). Antidote aims to be both a research platform for studying replication and consistency at the large scale, and an instrument for exploiting research results. The platform supports replication of CRDTs, in and between sharded (partitioned) data centres (DCs). The current stable version supports strong transactional consistency inside a DC, and causal transactional consistency between DCs. Ongoing research includes support for explicit consistency [23], for elastic version management, for adaptive replication, for partial replication, and for reconfigurable sharding.

SCALE Team

5. New Software and Platforms

5.1. Platforms

5.1.1. EventCloud

Participants : Iyad Alshabani, Maéva Antoine, Françoise Baude, Fabrice Huet, Laurent Pellegrino.

The *EventCloud* is an open source middleware that aims to act as a distributed datastore for data fulfiling the W3C RDF specification (http://www.w3.org/RDF/). It allows to store and retrieve quadruples (RDF triples with context) through SPARQL but also to manage events represented as quadruples. The *EventCloud* architecture is based on a structured P2P overlay network targetting high-performance elastic data processing. Consequently it aims to be deployed on infrastructures like grids, clouds, i.e. whose nodes acquisition and relinquishment can be dynamic and subject to a pay-per-use mode. Each node participating in the overlay networks constituting EventCloud instances, is responsible for managing the storage of subsets of the events, and helps in matching potential looked up events and disseminating them in a collaborative manner. As such, each node is also potentially an event broker responsible for managing subscriptions and routing notifications.

The *EventCloud* provides a high level publish-subscribe API where users can register their interests using SPARQL. When matching RDF data are added, subscribers are automatically notified. Recent work around the *EventCloud* has focused on efficient algorithms for managing subscription and notification.

5.1.2. BtrPlace

Participants : Fabien Hermenier, Vincent Kherbache, Ludovic Henrio.

BtrPlace is an open source virtual machine (VM) scheduler 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 constraints. Its extensibility is possible thanks to a composable core scheduling algorithm implemented using Constraint Programming. *BtrPlace* is currently bundled with a catalog of more than 20 constraints to address performance, fault tolerance, isolation, infrastructure management or energy efficiency concerns. It is currently used inside the FSN project OpenCloudWare (http://opencloudware.org/) and the European project DC4Cities (http://dc4cities.eu/).

This year we first put an emphase on *BtrPlace* dissemination. BtrPlace has been frequently released and it is now available online on a dedicated Web site (http://btrplace.org). To increase its visibility and to ease its integration, we decided to made *BtrPlace* directly available from the central repository of maven, the standard system to manage Java projects. Finally, BtrPlace has been registered on the *Agence pour la Protection des Programmes*.

5.1.3. OSA

Participants : Olivier Dalle.

OSA stands for Open Simulation Architecture. OSA (http://osa.inria.fr/) 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.

5.1.4. VerCors

Participants: Eric Madelaine, Ludovic Henrio, Bartlomiej Szejna, Nassim Jibai, Oleksandra Kulankhina, Siqi Li.

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 achieved this year a major version of the platform frontend, named VCE-v3, that is now distributed on our website, and used by some of our partners. It includes integrated graphical editors for GCM component architecture descriptions, UML classes, interfaces, and state-machines. The user diagrams can be checked using the recently published validation rules from [11]; then the corresponding GCM components can be executed using an automatic generation of the application ADL, and skeletons of Java files.

But VCE-v3 is using the Obeo-designer platform, which is a commercial product, and we have started a port to the newly available Sirius platform (http://eclipse.org/sirius/), with the goal to distribute the next major release of VCE, next year, under Sirius.

SPIRALS Team

5. New Software and Platforms

5.1. APISENSE®

Participants: Clive Ferret-Canape, Julien Duribreux, María Gómez Lacruz, Nicolas Haderer, Christophe Ribeiro, Romain Rouvoy [correspondant], Antoine Veuiller.

In 2014, new developments have been made on our APISENSE® distributed crowdsensing platform. APISENSE® now builds on a distributed infrastructure hosted in the Cloud that can better cope with scalability issues in the number of experiments, users, and volume of data to be collected in the wild. Data collected by participants can be exposed to applications and stakeholders via an Open Data API, which provides the ability to build realtime web applications from crowdsourced datasets. The APISENSE® mobile app, named BEE, can be freely downloaded from the Google Play Store. APISENSE® is part of the results of the PhD thesis of Nicolas Haderer [12] that was defended in November 2014. In 2014, APISENSE® has also been at the core of an industrial transfer action that aims at creating a spin-off company. The project is managed by Christophe Ribeiro and Romain Rouvoy. The project has been accepted (so-called qualification) in 2014 by the Inria investment fund IT-Translation. The project in supported by Direction Transfert & Innovation which will fund in 2015 the 1-year engineer contract of Christophe Ribeiro for maturing the project.

APISENSE® is a distributed platform dedicated to crowdsensing activities. Crowdsensing intends to leverage mobile devices to seamlessly collect valuable dataset for different categories of stakeholders. APISENSE® intends to be used in a wide variety of scientific and industrial domains, including network quality monitoring, social behavior analysis, epidemy predictions, emergency crisis support, open maps initiatives, debugging of applications in the wide. APISENSE® is composed of BEE.HIVE delivered as a *Platform-as-a-Service* (PaaS) to the stakeholders who can pilot and customize their own crowdsensing environment [108], and *Bee.mob* supporting participants with a mobile application to control the sensors to be shared with the rest of the world [96], [97]. The platform is used by the Metroscope consortium, an Internet scientific observatory initiative supported by Inria.

APISENSE® is at the core of the Inria ADT Focus CrowdLab project (see Section 8.2).

Web site: http://www.apisense.fr. Registered with the APP (*Agence pour la Protection des Programmes*) under reference IDDN.FR.001.080006.000.S.P.2013.000.10000 is pending. License: Proprietary.

5.2. FraSCAti

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

The novelty of 2014 consists in the development of the SOCLOUD platform for distributed multi-cloud systems. This platform has been defined in the context of the PhD thesis of Fawaz Paraiso [15] that was defended in June 2014. SOCLOUD is built on top of our existing FRASCATI platform. SOCLOUD enables to deploy, execute and manage an application that spans on several different cloud systems.

FRASCATI is a service-oriented component-based middleware platform implementing OASIS *Service Component Architecture* (SCA) specifications. The main originality of 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 [110]. 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. FRASCATI provides management tools like standalone, Web-based, and JMX-based graphical consoles and a dedicated scripting language for reconfiguring SCA applications. The whole 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. 292 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: [117], [116], [103], [104], [93], [92].

5.3. PowerAPI

Participants: Maxime Colmant, Loïc Huertas, Adel Noureddine, Romain Rouvoy [correspondant].

In 2014, new developments have been made on our POWERAPI library for monitoring energy in software systems. POWERAPI now includes an accurate power model, which supports both DFVS, hyper threads and turbo boost features of modern processors. This model has been assessed on acknowledged benchmarks (PARSEC, SPEC CPU, SPECjbb) and is used as a basis to estimate the power consumption of applications running in virtualised environments. Finally, POWERAPI has evolved towards a modular toolkit that can be used to build software-defined power meters supporting a wide range of input sources (*e.g.*, hardware performance counters, RAPL, PowerSpy). POWERAPI is part of the results of the PhD thesis of Adel Noureddine [14] that was defended in March 2014.

POWERAPI is a Scala-based library for monitoring energy in software systems. 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 [105], [88], [86], [87], energy hotspots and bugs detection [106], [107], and real-time distributed system monitoring.

POWERAPI is at the core of the Inria ADT eSurgeon project (see Section 8.2).

Web site: http://www.powerapi.org. Registered with the APP (Agence pour la Protection des Programmes) under reference IDDN.FR.001.400015.000.S.P.2012.000.10000. License: AGPL.

5.4. Saloon

Participants: Laurence Duchien, Clément Quinton, Daniel Romero Acero, Lionel Seinturier [correspondant].

SALOON is a framework for the selection and configuration of Cloud providers according to application requirements. The framework enables the specification of such requirements by defining ontologies. Each ontology provides a unified vision of provider offers in terms of frameworks, databases, languages, application servers and computational resources (i.e., memory, storage and CPU frequency). Furthermore, each provider is related to a Feature Model (FM) with attributes and cardinalities, which captures its capabilities. By combining the ontology and FMs, the framework is able to match application requirements with provider capabilities and select a suitable one. Specific scripts to the selected provider are generated in order to enable its configuration.

SALOON is the result of the PhD thesis of Clément Quinton [16] that was defended in October 2014. SALOON is partially developed in the context of the FP7 PaaSage project (see Section 8.3).

Registered with the APP (Agence pour la Protection des Programmes) under reference IDDN.FR.001.300002.000.S.P.2014.000.10800.

5.5. Spoon

Participants: Martin Monperrus [correspondant], Gérard Paligot, Nicolas Petitprez.

In 2014, SPOON has been at the core of an industrial transfer action that aims at creating a spin-off company. The project is managed by Nicolas Petitprez and Martin Monperrus. The project has been accepted (so-called qualification) in 2014 by the Inria investment fund IT-Translation. The project in supported by Direction Transfert & Innovation which will fund in 2015 the 1-year engineer contract of Nicolas Petitprez for maturing the project. As an open source project Spoon has attracted new contributors in 2014. The Spoon development team is now composed of 8 active members, including 4 that are not at all related to Inria. Second, Spoon now supports analyzing and transforming Java 7 code, which is the now the dominant version of Java. Third, Spoon is the technical foundation of five important papers published in 2014. To sum up, year 2014 was a major year for warming up the Spoon project. Thanks to the support of Inria through the ADT, year 2015 is expected to be as vibrant and rich.

SPOON is a library for analyzing and transforming Java source code [76] [109]. SPOON provides a core API and associated tools for static analysis and generative programming within the Java 5+ environment. SPOON must be seen as a basis to ensure Software Quality through code validation and generation. It can be used in the software development process during the validation phase, as well as for engineering and re-engineering software. The first key point of SPOON is to provide a well-typed and comprehensive AST API which is designed to facilitate analysis and transformation work for programmers. Scanners and processors allow the programmer to implement various program traversal strategies on the Java program. Also, the program representation is built with a well-known and well-tested open source Java compiler: the Eclipse JDT compiler, which ensures the support of the latest Java features. The second key point of SPOON is to provide a pure Java API to specify program transformations using a well-typed generative programming technique (called Spoon Templates). By using well-typed templates, SPOON makes programming of transformations easier and safer for the end-user programmers.

SPOON is at the core of the Inria ADT Spoon3R project (see Section 8.1).

Web site: http://spoon.gforge.inria.fr. Registered with the APP (Agence pour la Protection des Programmes) under reference IDDN.FR.001.070037.000.S.P.2007.000.10600. License: CeCILL-C.

WHISPER Team

5. New Software and Platforms

5.1. Platforms

5.1.1. Coccinelle

Our recent research is in the area of code manipulation tools for C code, particularly targeting Linux kernel code. This work has led to the Coccinelle tool that we are continuing to develop. Coccinelle serves both as a basis for our future research and the foundation of our interaction with the Linux developer community.

The need to find patterns of code, and potentially to transform them, is pervasive in software development. Examples abound. When a bug is found, it is often fruitful to see whether the same pattern occurs elsewhere in the code. For example, the recent Heartbleed bug in OpenSSL partly involves the same fragment of code in two separate files. [0] Likewise, when the interface of an API function changes, all of the users of that function have to be updated to reflect the new usage requirements. This generalizes to the case of code modernization, in which a code base needs to be adapted to a new compiler, new libraries, or a new coding standards. Finding patterns of code is also useful in code understanding, *e.g.*, to find out whether a particular function is ever called with a particular lock held, and in software engineering research, *e.g.*, to understand the prevalence of various kinds of code structures, which may then be correlated with other properties of the software. For all of these tasks, there is a need for an easy to use tool that will allow developers to express patterns and transformations that are relevant to their source code, and to apply these patterns and transformations to the code efficiently and without disrupting the overall structure of the code base.

To meet these needs, we have developed the Coccinelle program matching and transformation tool for C code. Coccinelle has been under development for over 7 years, and is mature software, available in a number of Linux distributions (Ubuntu, Debian, Fedora, etc.). Coccinelle allows matching and transformation rules to be expressed in terms of fragments of C code, more precisely in the form of a *patch*, in which code to add and remove is highlighted by using + and −, respectively, in the leftmost column, and other, unannotated, code fragments may be provided to describe properties of the context. The C language is extended with a few operators, such as metavariables, for abstracting over subterms, and a notion of positions, which are useful for reporting bugs. The pattern matching rules can interspersed with rules written in Python or OCaml, for further expressiveness. The process of matching patterns against the source code furthermore takes into account some semantic information, such as the types of expressions and reachability in terms of a function's (intraprocedural) control-flow graph, and thus we refer to Coccinelle matching and transformation specifications as *semantic patches*.

Coccinelle was originally motivated by the goal of modernizing Linux 2.4 drivers for use with Linux 2.6, and was originally validated on a collection of 60 transformations that had been used in modernizing Linux 2.4 drivers [8]. Subsequent research involving Coccinelle included a formalization of the logic underlying its implementation [1] and a novel mechanism for identifying API usage protocols [50]. More recently, Coccinelle has served as a practical and flexible tool in a number of research projects that somehow involve code understanding or transformation. These include identifying misuses of named constants in Linux code [52], extracting critical sections into procedures to allow the implementation of a centralized locking service [58], generating a debugging interface for Linux driver developers [31], detecting resource release omission faults in Linux and other infrastructure software [68], and understanding the structure of device driver code in our current DrGene project [70].

[0] http://git.openssl.org/gitweb/?p=openssl.git;a=commitdiff;h=96db902

Throughout the development of Coccinelle, we have also emphasized contact with the developer community, particularly the developers of the Linux kernel. We submitted the first patches to the Linux kernel based on Coccinelle in 2007. Since then, over 2000 patches have been accepted into the Linux kernel based on the use of Coccinelle, including around 700 by around 90 developers from outside our research group. Over 40 semantic patches are available in the Linux kernel source code itself, with appropriate infrastructure for developers to apply these semantic patches to their code within the normal `make` process. Many of these semantic are also included in a 0-day build-testing system for Linux patches maintained by Intel. [0] Julia Lawall was invited to the Linux Kernel Summit as a core attendee (invitation only) in 2010 and 2014, and has been invited to the internal 2014 SUSE Labs Conference. She has also presented Coccinelle at developer events such as LinuxCon Europe, Kernel Recipes (Paris), FOSDEM (Brussels), and RTWLS, and has supervised a summer intern financed by the Linux Foundation, as part of the GNOME Foundation's Outreach Program for Women.

Finally, we are aware of several companies that use Coccinelle for modernizing code bases. These include Metaware in Paris, with whom we have had a 5-month contract in 2013-2014 for the customization and maintenance of Coccinelle. We hope to be able to organize other such contracts in the future.

5.1.2. Better Linux

Over the past few years, Julia Lawall and Gilles Muller have designed and developed of a number of tools such as Coccinelle, Diagnosys [31] [30] and Hector [68], to improve the process of developing and maintaining systems code. The BtrLinux action aims to increase the visibility of these tools, and to highlight Inria's potential contributions to the open source community. We will develop a web site `https://BtrLinux.inria.fr`, to centralize the dissemination of the tools, collect documentation, and collect results. This action is supported by Inria by the means of a young engineer (ADT), Quentin Lambert. In the case of Coccinelle, we will focus on enhancing its visibility and its dissemination, by using it to find and fix faults in Linux kernel code, and by submitting the resulting patches to the Linux maintainers. We now present the other tools considered in the BtrLinux action in more detail.

Diagnosys is a hybrid static and dynamic analysis tool that first collects information about Linux kernel APIs that may be misused, and then uses this information to generate wrapper functions that systematically log at runtime any API invocations or return values that may reflect such misuse. A developer can then use a specific make-like command to build an executable driver that transparently uses these wrapper functions. At runtime, the wrappers write log messages into a crash resilient region of memory that the developer can inspect after any crash. Diagnosys is complementary to Coccinelle in the kind of information that it provides to developers. While Coccinelle directly returns a report for every rule match across the code base, often including false positives that have to be manually isolated by the developer, Diagnosys only reports on conditions that occur in the actual execution of the code. Diagnosys thus produces less information, but the information produced is more relevant to the particular problem currently confronting the developer. As such, it is well suited to the case of initial code development, where the code is changing frequently, and the developer wants to debug a specific problem, rather than ensuring that the complete code base is fault free. Diagnosys is a complete functioning system, but it needs to be kept up to date with changes in the kernel API functions. As part of the BtrLinux action, we will regularly run the scripts that collect information about how to create the wrappers, and then validate and make public the results.

Hector addresses the problem of leaking resources in error-handling code. Releasing resources when they are no longer needed is critical, so that adequate resources remain available over the long execution periods characteristic of systems software. Indeed, when resource leaks accumulate, they can cause unexpected resource unavailability, and even single leaks can put the system into an inconsistent state that can cause crashes and open the door to possible attacks. Nevertheless, developers often forget to release resources, because doing so often does not make any direct contribution to a program's functionality. A major challenge in detecting resource-release omission faults is to know when resource release is required. Indeed, the C language does not provide any built-in support for resource management, and thus resource acquisition and release are typically implemented using ad hoc operations that are, at best, only known to core developers.

[0] E.g., http://comments.gmane.org/gmane.linux.kernel.kbuild/269

Previous work has focused on mining sequences of such functions that are used frequently across a code base, [43], [56] but these approaches have very high rates of false negatives and false positives. [53] We have proposed Hector, a static analysis tool that finds resource-release omission faults based on inconsistencies in the operations performed within a single function, rather than on usage frequency. This strategy allows Hector to have a low false positive rate, of 23% in our experiments, while still being able to find hundreds of faults in Linux and other systems.

Hector was developed as part of the PhD thesis of Suman Saha and was presented at DSN 2013, where it received the William C. Carter award for the best student paper. Hector is complementary to Coccinelle, in that it has a more restricted scope, focusing on only one type of fault, but it uses a more precise static analysis, tailored for this type of fault, to ensure a low false positive rate. Hector, like Coccinelle, is also complementary to Diagnosys, in that it exhaustively reports on faults in a code base, rather than only those relevant to a particular execution, and is thus better suited for use by experienced developers of relatively stable software. Over 70 patches have been accepted into Linux based on the results of Hector. The current implementation, however, is somewhat in a state of disarray. As part of the BtrLinux action, we will first return the code to working condition and then actively use it to find faults in Linux. Based on these results, we will either submit appropriate patches to the Linux developers or notify the relevant developer when the corresponding fix is not clear.

ALGORILLE Project-Team

5. New Software and Platforms

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. Implementing parallel models

Several software platforms have served us to implement and promote our ideas in the domain of coarse grained computation and application structuring.

5.2.1. ORWL and P99

Participants: Jens Gustedt, Stéphane Vialle [External collaborator, SUPELEC], Mariem Saied.

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

5.2.2. parXXL

Participants: Jens Gustedt, Stéphane Vialle [External collaborator, SUPELEC].

ParXXL is a library for large scale computation and communication that executes fine grained algorithms on coarse grained architectures (clusters, grids, mainframes). It has been 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

5.2.3. musl

Participant: Jens Gustedt.

musl is a re-implementation of the C library as it is described by the C and POSIX standards. It is *lightweight*, *fast*, *simple*, *free*, and strives to be correct in the sense of standards-conformance and safety. Musl is production quality code that is mainly used in the area of embedded device. It gains more market share also in other area, *e.g.* there are now Linux distributions that are based on musl instead of Gnu LibC.

In 2014, we have added an implementation of the new thread interface that had been defined in the recent C11 standard.

5.3. Parallel developments for numerical scientific application

Participant: Sylvain Contassot-Vivier.

The RAD2D/RAD3D software are co-developed with Fatmir Asllanaj, full researcher in physics at the LEMTA Laboratory, in the context of an inter-disciplinary collaboration. The object of those software is to solve and compute the radiative-transfer equation by using the finite volume method. As the amount of computations induced is very large, the resort to parallelism is mandatory [9], [15]. By its complexity and similarity with a large proportion of scientific applications, this real case application is a fully pertinent test-case for the parallel techniques and schemes we have designed in our team. Those software are not open-source and, by the way, are still in development state.

5.4. 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.

5.5. SimGrid

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.

5.5.1. Core distribution

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

SimGrid has an active user community of more than one hundred members, and is available under GPLv3 from http://simgrid.gforge.inria.fr/. One third of the source code is devoted to about 12000 unit tests and 500 full integration tests. These tests are run for each commit for 4 package configurations and on 4 operating systems thanks to the Inria continuous integration platform.
Software classification: A-5, SO-4, SM-4, EM-4, SDL-5, DA-4, CD-4, MS-4, TPM-4.

5.5.2. SimGridMC

Participants: Martin Quinson, Marion Guthmuller, Gabriel Corona.

SimGridMC is a module of SimGrid that can be used to formally assess any distributed system that can be simulated within SimGrid. It explores all possible message interleavings searching for states violating the provided properties. We recently added the ability to assess liveness properties over arbitrary C codes, thanks to a system-level introspection tool that provides a finely detailed view of the running application to the model checker. This can for example be leveraged to verify arbitrary MPI code written in C.
Software classification: A-3-up, SO-4, SM-3-up, EM-3-up, SDL-5, DA-4, CD-4, MS-4, TPM-4.

5.5.3. SCHIaaS

Participants: Julien Gossa [External collaborator, SUPELEC], Stéphane Genaud [External collaborator, SUPELEC], Rajni Aron.

The *Simulation of Clouds, Hypervisor and IaaS* (SCHIaaS) is an extension of SimGrid that can be used to comprehensively simulate clouds, from the hypervisor/system level, to the IaaS/administrator level. The hypervisor level includes models about virtualization overhead and VMs operations like boot, start, suspend, migrate, and network capping. The IaaS level includes models about instances management like image storage and deployment and VM scheduling. This extension allows to fully simulate any cloud infrastructure, whatever the hypervisor or the IaaS manager. This can be used by both cloud administrators to dimension and tune clouds, and cloud users to simulate cloud applications and assess provisioning strategies in term of performances and cost.
Software classification: A-3-up, SO-3, SM-2-up, EM-2-up, SDL-2, DA-4, CD-4, MS-4, TPM-4.

5.6. Kadeploy

Participants: Luc Sarzyniec, Stéphane Martin, Emmanuel Jeanvoine, Lucas Nussbaum [correspondant].

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.

5.7. XPFlow

Participants: Tomasz Buchert, Lucas Nussbaum [correspondant].

XPFlow is an implementation of a new, workflow-inspired approach to control experiments involving large-scale computer installations. Such systems pose many difficult problems to researchers due to their complexity, their numerous constituents and scalability problems. The main idea of the approach consists in describing the experiment as a workflow and execute it using achievements of Business Process Management (BPM), workflow management techniques and scientific workflows. The website of XPFlow is http://xpflow.gforge.inria.fr/. XPFlow was featured in a tutorial during Grid'5000 Spring School 2014.
Software classification: A-2-up, SO-3-up, SM-2-up, EM-3-up, SDL-2-up, DA-4, CD-4, MS-4, TPM-4.

5.8. Grid'5000 testbed

Participants: Luc Sarzyniec, Jérémie Gaidamour, Arthur Garnier, Clément Parisot, Emmanuel Jeanvoine, Émile Morel, Lucas Nussbaum [correspondant].

Grid'5000 (http://www.grid5000.fr) is a scientific instrument designed to support experiment-driven research in all areas of computer science related to parallel, large-scale or distributed computing and networking. It gathers 10 sites, 25 clusters, 1200 nodes, for a total of 8000 cores. It provides its users with a fully reconfigurable environment (bare metal OS deployment with Kadeploy, network isolation with KaVLAN) and a strong focus on enabling high-quality, reproducible experiments.
The AlGorille team contributes to the design of Grid'5000, to the administration of the local Grid'5000 site in Nancy, and to the design and development of Kadeploy (in close cooperation with the Grid'5000 technical team). The AlGorille engineers also administer *Inria Nancy – Grand Est*'s local production cluster, named *Talc*, leveraging the experience and tools from Grid'5000.

Software classification: A-5, SO-4, SM-4, EM-4, SDL-N/A, DA-4, CD-4, MS-4, TPM-4.

ALPINES Project-Team

5. New Software and Platforms

5.1. Platforms

5.1.1. FreeFem++, http://www.freefem.org/ff++/

FreeFem++ is a PDE solver based on a flexible language that allows a large number of problems to be expressed (elasticity, fluids, etc) with different finite element approximations on different meshes. There are more than 2000 users, and on the mailing list there are 430 members. Among those, we are aware of at least 10 industrial companies, 8 french companies and 2 non-french companies. It is used for teaching at Ecole Polytechnique, Ecole Centrale, Ecole des Ponts, Ecole des Mines, University Paris 11, University Paris Dauphine, La Rochelle, Nancy, Metz, Lyon, etc. Outside France, it is used for example at universities in Japan (Tokyo, Kyoto, Hiroshima, there is a userguide FreeFem++ in japan), Spain (Sevilla, BCAM, userguide available in spanish), UK (Oxford), Slovenia, Switzerland (EPFL, ETH), China. For every new version, there are 350 regression tests, and we provide a rapid correction of reported bugs. The licence of FreeFem++ is LGPL.

5.1.2. Library for preconditioned iterative methods

In the project-team we develop a library that integrates the direction preserving and low rank approximation preconditioners for both approached factorizations and domain decomposition like methods. It will be available through `FreeFem++` and also as a stand alone library, and we expect to have one version of this library available in 2014.

5.1.3. HPDDM, https://github.com/hpddm

HPDDM is an efficient implementation of various domain decomposition methods (DDM) such as one- and two-level Restricted Additive Schwarz methods, the Finite Element Tearing and Interconnecting (FETI) method, and the Balancing Domain Decomposition (BDD) method. These methods can be enhanced with deflation vectors computed automatically by the framework using methods developed by members of the team:

- Generalized Eigenvalue problems on the Overlap (GenEO), an approach first introduced in the PhD of Nicole Spillane.
- local Dirichlet-to-Neumann operators, an approach first introduced in a paper by Nataf et al. and recently revisited by Conen et al.

This code has been proven to be efficient for solving various elliptic problems such as scalar diffusion equations, the system of linear elasticity, but also frequency domain problems like the Helmholtz equation. A comparison with modern multigrid methods can be found in the thesis of Pierre Jolivet.

HPDDM is a header-only library written in C++11 with MPI and OpenMP for parallelism. While its interface relies on plain old data objects, it requires a modern C++ compiler: g++ 4.7.3 and above, clang++ 3.3 and above, icpc 15.0.0.090 and above. HPDDM has to be linked against BLAS and LAPACK (as found in OpenBLAS, in the Accelerate framework on OS X, in IBM ESSL, or in Intel MKL) as well as a direct solver like MUMPS, SuiteSparse, MKL PARDISO, or PaStiX. At compilation, just define before including HPDDM.hpp one of these preprocessor macros MUMPSSUB, SUITESPARSESUB, MKL_PARDISOSUB, or PASTIXSUB (resp. DMUMPS, DSUITESPARSE, DMKL_PARDISO, or DPASTIX) to use the corresponding solver inside each subdomain (resp. for the coarse operator). Additionally, an eigenvalue solver is recommended. There is an existing interface to ARPACK. Other (eigen)solvers can be easily added using the existing interfaces. For building robust two-level methods, an interface with a discretization kernel like FreeFem++ or Feel++ is also needed. It can then be used to provide, for example, elementary matrices, that the GenEO approach requires. As such HPDDM is not an algebraic solver, unless only looking at one-level methods. Note that for substructuring methods, this is more of a limitation of the mathematical approach than of HPDDM itself.

AVALON Project-Team

5. New Software and Platforms

5.1. BitDew/Active Data

Participants: Gilles Fedak [correspondant], Anthony Simonet.

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 extension focuses on Active Data, which is a data-centric and event-driven programming model combined with a runtime environment, which allows to expose and manage data set life cycle. Active Data strength is to facilitate the development of applications that handle dynamic data sets distributed on heterogeneous systems and infrastructures.

5.2. DIET

Participants: Daniel Balouek Thomert, Eddy Caron [correspondant], Frédéric Desprez, Maurice Faye, Arnaud Lefray, Guillaume Verger, 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.

DIET provide a support for Cloud architecture. and it takes 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.

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. By the way, we have added the SSL support into the DIET communication layer. We have worked to show how to securely use public cloud storage without taking the risk of losing confidentiality of data stored on them.

We have started a work to design a plug-in schedulers into DIET to deal with energy management. Using this scheduler we have obtain a significatif gain close to 25% with a minor weakening of performance (6%). Moreover we have experimented some dynamic resources management through DIET based on the energy criteria.

5.3. Sam4c

Participants: Eddy Caron, Arnaud Lefray [correspondant], Jonathan Rouzaud-Cornabas.

Sam4C (https://gforge.inria.fr/projects/sam4c/) -Security-Aware Models for Clouds- is a graphical and textual editor to model Cloud applications (as virtual machines, processes, files and communications) and describe its security policy. Sam4C is suitable to represent any static application without deadline or execution time such as n-tiers or parallel applications. This editor is generated in Java from an EMF -Eclipse Modeling Framework- metamodel to simplify any modifications or extensions. The application model and the associated security policy are compiled in a single XML file which serves as input for an external Cloud security-aware scheduler. Alongside with this editor, Cloud architecture models and provisioning algorithms are provided for simulation (in the current version) or real deployments (in future versions). During this step of development this software is private and available only for Seed4C project members. The design of Sam4c is a joint effort with INSA Centre Val de Loire.

5.4. SimGrid

Participants: 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/.

5.5. HLCMi, L^{2}C, & Gluon++

Participants: Hélène Coullon, Vincent Lanore, Christian Perez [correspondant], Jérôme Richard.

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/L^{2}C and HLCM/Charm++ (known as Gluon++).

L^2C (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^2C does not offer network transparency neither language transparency. Instead, L^2C lets the user choose between various kinds of interactions between components, some with ultra low overhead and others that support network transport. L^2C is extensible as additional interaction kinds can be added quite easily. L^2C currently supports C++, FORTRAN 2013, MPI and CORBA interactions.

Gluon++ (http://hlcm.gforge.inria.fr) is a thin component model layer added on top of Charm++ (http://charm.cs.uiuc.edu/). It defines chare components as a Charm++ chare with minimal metadata, C++ components as a C++ class with minimal metadata, (asynchronous) entry method calls between components, and plain C++ method calls between components.

L^2C 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++.

5.6. Execo

Participants: Matthieu Imbert [correspondant], Laurent Pouilloux.

`Execo`(http://execo.gforge.inria.fr) is a Python library designed for rapid prototyping of experiments on distributed systems, automatization of system administration tasks (such as deployment and configuration of distributed middleware), and creation of reproducible experiments scripts. It allows easy and asynchronous management of thousands of local or remote unix processes and offers tools for easy usage of the Grid'5000 platform services.

`Execo` currently has more than 20 users in and outside the AVALON team, who rely on it to automate experimental workflows. It was used to develop one of the two contenders who won the 2014 Grid'5000 Large Scale Deployment Challenge. It is used as a building block in the Grid'5000 metrology service and has been used to produce experimental results involved in numerous papers and reports.

It is distributed under GPLv3 and it is made of 7200 lines of code.

5.7. Kwapi

Participants: Laurent Lefèvre [correspondant], François Rossigneux, Jean-Patrick Gelas, Laurent Pouilloux.

Kwapi (https://launchpad.net/kwapi) is a software framework dealing with energy monitoring of large scale infrastructures through heterogeneous energy sensors. Kwapi has been designed inside the FSN XLCloud project for Openstack infrastructures. Through the support of Hemera Inria project, kwapi has been extended and deployed in production mode to support easy and large scale energy profiling of the Grid5000 resources.

5.8. Platforms

5.8.1. Grid'5000

Participants: Frédéric Desprez, Simon Delamare, Laurent Lefèvre, David Loup, Christian Perez, Marc Pinhède, Laurent Pouilloux.

The GRID'5000 experimental platform (http://www.grid5000.fr) is a scientific instrument to support computer science research related to distributed systems, including parallel processing, high performance computing, cloud computing, operating systems, peer-to-peer systems and networks. It is distributed on 10 sites in France and Luxembourg, including Lyon. GRID'5000 is a unique platform as it offers to researchers many and varied hardware resources and a complete software stack to conduct complex experiments, ensure reproducibility and ease understanding of results.

Not only GRID'5000 is heavily used for Avalon research, but several team members are also involved in GRID'5000 direction:

- Frédéric Desprez is leading the "Groupement d'Intérêt Scientifique Groupement Grille 5K" which drives GRID'5000.
- Laurent Lefèvre is responsible of the GRID'5000 Lyon platform and member of the GRID'5000 direction committee.
- Christian Perez is leading the Hemera initiative (https://www.grid5000.fr/Hemera) and he is a member of the GRID'5000 direction committee.
- Simon Delamare is the plateform's operational manager.

Avalon also provides an important effort for Grid'5000 operation and development by hosting several enginneers belonging to Grid'5000 technical team (Marc Pinhède, David Loup) or HEMERA IPL (Laurent Pouilloux).

HIEPACS Project-Team

5. New Software and Platforms

5.1. Introduction

We describe in this section the software that we are developing. The first list 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 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

Participant: Emmanuel Agullo [corresponding member].

MaPHyS (Massivelly Parallel Hybrid Solver) is a software package that implements a parallel linear solver coupling 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 .

MaPHyS can be found at http://maphys.gforge.inria.fr.

5.3. PaStiX

Participant: Pierre Ramet [corresponding member].

Complete and incomplete supernodal sparse parallel factorizations.

PaStiX (Parallel Sparse matriX package) is a scientific library that provides a high performance parallel solver for very large sparse linear systems based on block direct and block ILU(k) iterative methods. Numerical algorithms are implemented in single or double precision (real or complex): LLt (Cholesky), LDLt (Crout) and LU with static pivoting (for non symmetric matrices having a symmetric pattern).

The PaStiX library uses the graph partitioning and sparse matrix block ordering package Scotch. PaStiX is based on an efficient static scheduling and memory manager, in order to solve 3D problems with more than 50 million of unknowns. The mapping and scheduling algorithm handles a combination of 1D and 2D block distributions. This algorithm computes an efficient static scheduling of the block computations for our supernodal parallel solver which uses a local aggregation of contribution blocks. This can be done by taking into account very precisely the computational costs of the BLAS 3 primitives, the communication costs and the cost of local aggregations. We also improved this static computation and communication scheduling algorithm to anticipate the sending of partially aggregated blocks, in order to free memory dynamically. By doing this, we are able to reduce the aggregated memory overhead, while keeping good performance.

Another important point is that our study is suitable for any heterogeneous parallel/distributed architecture when its performance is predictable, such as clusters of multicore nodes. In particular, we now offer a high performance version with a low memory overhead for multicore node architectures, which fully exploits the advantage of shared memory by using an hybrid MPI-thread implementation.

Direct methods are numerically robust methods, but the very large three dimensional problems may lead to systems that would require a huge amount of memory despite any memory optimization. A studied approach consists in defining an adaptive blockwise incomplete factorization that is much more accurate (and numerically more robust) than the scalar incomplete factorizations commonly used to precondition iterative solvers. Such incomplete factorization can take advantage of the latest breakthroughs in sparse direct methods and particularly should be very competitive in CPU time (effective power used from processors and good scalability) while avoiding the memory limitation encountered by direct methods.

PaStiX is publicly available at http://pastix.gforge.inria.fr under the Inria CeCILL licence.

5.4. HIPS

Participant: Pierre Ramet [corresponding member].

Multilevel method, domain decomposition, Schur complement, parallel iterative solver.

HIPS (Hierarchical Iterative Parallel Solver) is a scientific library that provides an efficient parallel iterative solver for very large sparse linear systems.

The key point of the methods implemented in HIPS is to define an ordering and a partition of the unknowns that relies on a form of nested dissection ordering in which cross points in the separators play a special role (Hierarchical Interface Decomposition ordering). The subgraphs obtained by nested dissection correspond to the unknowns that are eliminated using a direct method and the Schur complement system on the remaining of the unknowns (that correspond to the interface between the sub-graphs viewed as sub-domains) is solved using an iterative method (GMRES or Conjugate Gradient at the time being). This special ordering and partitioning allows for the use of dense block algorithms both in the direct and iterative part of the solver and provides a high degree of parallelism to these algorithms. The code provides a hybrid method which blends direct and iterative solvers. HIPS exploits the partitioning and multistage ILU techniques to enable a highly parallel scheme where several subdomains can be assigned to the same process. It also provides a scalar preconditioner based on the multistage ILUT factorization.

HIPS can be used as a standalone program that reads a sparse linear system from a file ; it also provides an interface to be called from any C, C++ or Fortran code. It handles symmetric, unsymmetric, real or complex matrices. Thus, HIPS is a software library that provides several methods to build an efficient preconditioner in almost all situations.

HIPS is publicly available at http://hips.gforge.inria.fr under the Inria CeCILL licence.

5.5. MetaPart

Participant: Aurélien Esnard [corresponding member].

MetaPart is a library that adresses the challenge of (dynamic) load balancing for emerging complex parallel simulations, such as multi-physics or multi-scale coupling applications. First, it offers a uniform API over state-of-the-art (hyper-) graph partitioning software packages such as Scotch, PaToH, METIS, Zoltan, Mondriaan, etc. etc. Based upon this API, it provides a framework that facilitates the development and the evaluation of high-level partitioning methods, such as MxN repartitioning or coupling-aware partitionining (co-partitioning).

The framework is publicy available at Inria Gforge: http://metapart.gforge.inria.fr.

5.6. MPICPL

Participant: Aurélien Esnard [corresponding member].

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.

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

5.7. ScalFMM

Participant: Olivier Coulaud [corresponding member].

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

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 are two main parts: 1) the management of the octree and the parallelization of the method ; 2) the kernels. This new architecture allows us to easily add new FMM algorithm or kernels and new paradigm of parallelization. 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.

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 at 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.

The ScalFMM package is available at http://scalfmm.gforge.inria.fr

5.8. ViTE

Participant: Mathieu Faverge [corresponding member].

Visualization, Execution trace

ViTE is a trace explorer. It is a tool made to visualize execution traces of large parallel programs. It supports Pajé, a trace format created by Inria Grenoble, and OTF and OTF2 formats, developed by the University of Dresden and allows the programmer a simpler way to analyse, debug and/or profile large parallel applications. It is an open source software licenced under CeCILL-A.

The ViTE software is available at http://vite.gforge.inria.fr and has been developed in collaboration with the Inria Bordeaux - Sud-Ouest SED team, Telecom SudParis and Inria Grenoble.

In the same context we also contribute to the EZtrace and GTG libraries in collaboration with F. Trahay from Telecom SudParis. EZTrace (http://eztrace.gforge.inria.fr) is a tool that aims at generating automatically execution trace from HPC programs. It generates execution trace files thanks to the GTG library (http://gtg.gforge.inria.fr) that can be later interpreted by visualization tools such as ViTE.

5.9. Other software

For the materials physics applications, a lot of development will be done in the context of ANR projects (NOSSI and OPTIDIS, see Section 4.1) in collaboration with LaBRI, CPMOH, IPREM, EPFL and with CEA Saclay and Bruyère-le-Châtel.

- **FAST**
  Participant: Olivier Coulaud [corresponding member].

  FAST is a linear response time dependent density functional program for computing the electronic absorption spectrum of molecular systems. It uses an O(N^3) linear response method based on finite numerical atomic orbitals and deflation of linear dependence in atomic orbital product space. This version is designed to work with data produced by the SIESTA DFT code. The code produces as principal output a numerical absorption spectrum (complex part of the polarisability, loosely called the polarisability below) and a list of transition energies and oscillator strengths deduced from fitting Lorentzians to the numerical spectrum. Considering the absence of hybrid functionals in SIESTA and that concerning calculation of spectra, generalized gradient Hamiltonians are not usually considered to be notably better than the local density approximation, the present release of FAST works only with LDA, which despite its limitations, has provided useful results on the systems to which the present authors have applied it. The FAST library is available at http://people.bordeaux.inria.fr/coulaud/Softwares/FAST/index.html.

- **OptiDis**
  Participant: Olivier Coulaud [corresponding member].

  OptiDis is a new code for large scale dislocation dynamics simulations. Its aim is to simulate real life dislocation densities (up until 5.10^{22} dislocations/m^{-2}) in order to understand plastic deformation and study strain hardening. The main application is to observe and understand plastic deformation on irradiated zirconium. Zirconium alloys is the first containment barrier against the dissemination of radioactive elements. More precisely, with neutron irradiated zirconium alloys we are talking of channeling mechanism, which means to stick with the reality,more than tens of thousands of induced loops so 10^8 degrees of freedom in the simulation.

  The code is based on Numodis code developed at CEA Saclay and the ScalFMM library developed in our Inria project. The code is written in C++ language and using the last features of C++11. One of the main aspects is the hybrid parallelism MPI/OpenMP that gives the software the ability to scale on large cluster while the computation load rises. In order to achieve that, we use different levels of parallelism. First of all, the simulation box is spread over MPI processes, we then use a thinner level for threads, dividing the domain using an Octree representation. All theses parts are driven by the ScalFMM library. On the last level our data are stored in an adaptive structure absorbing dynamic of this kind of simulation and handling well task parallelism.

The two following packages are mainly designed and developed in the context of a US initiative led by ICL and to which we closely collaborate through the associate team MORSE.

- **PLASMA**
  Participant: Mathieu Faverge [corresponding member].

  The PLASMA (Parallel Linear Algebra for Scalable Multi-core Architectures) project aims at addressing the critical and highly disruptive situation that is facing the Linear Algebra and High Performance Computing community due to the introduction of multi-core architectures.

  The PLASMA ultimate goal is to create software frameworks that enable programmers to simplify the process of developing applications that can achieve both high performance and portability across a range of new architectures.

  The development of programming models that enforce asynchronous, out of order scheduling of operations is the concept used as the basis for the definition of a scalable yet highly efficient software framework for Computational Linear Algebra applications.

The PLASMA library is available at http://icl.cs.utk.edu/plasma.

- **PaRSEC/DPLASMA**
  Participant: Mathieu Faverge [corresponding member].

  PaRSEC Parallel Runtime Scheduling and Execution Controller, is a generic framework for architecture aware scheduling and management of micro-tasks on distributed many-core heterogeneous architectures. Applications we consider can be expressed as a Direct Acyclic Graph of tasks with labeled edges designating data dependencies. DAGs are represented in a compact problem-size independent format that can be queried on-demand to discover data dependencies in a totally distributed fashion. PaRSEC assigns computation threads to the cores, overlaps communications and computations and uses a dynamic, fully-distributed scheduler based on architectural features such as NUMA nodes and algorithmic features such as data reuse.

  The framework includes libraries, a runtime system, and development tools to help application developers tackle the difficult task of porting their applications to highly heterogeneous and diverse environments.

  DPLASMA (Distributed Parallel Linear Algebra Software for Multicore Architectures) is the leading implementation of a dense linear algebra package for distributed heterogeneous systems. It is designed to deliver sustained performance for distributed systems where each node featuring multiple sockets of multicore processors, and if available, accelerators like GPUs or Intel Xeon Phi. DPLASMA achieves this objective through the state of the art PaRSEC runtime, porting the PLASMA algorithms to the distributed memory realm.

  The PaRSEC runtime and the DPLASMA library are available at http://icl.cs.utk.edu/parsec.

5.10. Platforms

5.10.1. PlaFRIM: an experimental parallel computing platform

PLAFRIM is an experimental platform for research in modeling, simulations and high performance computing. This platform has been set up from 2009 under the leadership of Inria Bordeaux Sud-Ouest in collaboration with computer science and mathematics laboratories, respectively Labri and IMB with a strong support in the region Aquitaine.

It aggregates different kinds of computational resources for research and development purposes. The latest technologies in terms of processors, memories and architecture are added when they are available on the market. It is now more than 1,000 cores (excluding GPU and Xeon Phi) that are available for all research teams of Inria Bordeaux, Labri and IMB. This computer is in particular used by all the engineers who work in HiePACS and are advised by F. Rue from the SED.

The PlaFRIM platform initiative is coordinated by O. Coulaud.

KerData Project-Team

5. New Software and Platforms

5.1. Major Software

5.1.1. BlobSeer

Participants: Loïc Cloatre, 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*. This year we have mainly focused on the deployment in production of the BlobSeer software on IBM's cluster at Montpellier, in the context of the ANR MapReduce project. To this end, several bugs were solved, and several optimizations were brought to the communication layer of BlobSeer. To showcase the benefits of BlobSeer on this platform we focused on the Terasort benchmark. Currently, preliminary tests on Grid5000 with this benchmark show that BlobSeer performs better than HDFS for block sizes lower than 2 MB. We have also improved the continuous integration process of BlobSeer by deploying daily builds and automatic tests on Grid5000.

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) the OpenNebula IaaS cloud toolkit 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 *Technology Research Action* (ADT, *Action de recherche technologique*) started in November 2012 for two years, aiming at robustifying the BlobSeer software and making it a safely distributable product. This project is funded by Inria *Technological Development Office* (D2T, *Direction du Développement Technologique*). Loïc Cloatre has been hired as a senior engineer for the second year of this project, as a successor of Zhe Li, starting in February 2014.

5.1.2. Damaris

Participants: Matthieu Dorier, Orçun Yildiz, Lokman Rahmani, Shadi Ibrahim, Gabriel Antoniu.

Contact: Gabriel Antoniu.

Presentation: Damaris is a middleware for multicore SMP nodes enabling them to handle data transfers for storage and visualization efficiently. 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). Damaris enables efficient asynchronous I/O, hiding all I/O related overheads such as data compression and post-processing, as well as direct (*in-situ*) interactive visualization of the generated data. Version 1.0 was released in November 2014 and enables other approaches such as the use of dedicated nodes instead of dedicated cores.

Users: Damaris has been preliminarily evaluated at NCSA/UIUC (Urbana-Champaign, IL, USA) with the CM1 tornado simulation code. CM1 is one of the target applications of the Blue Waters supercomputer in production at, 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 four large-scale simulations (CM1, OLAM, Nek5000, GTC).

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

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

Status: This software is available on Inria's forge and registered with APP. Registration of the latest version with APP is in progress.

5.2. New Software

5.2.1. Omnisc'IO

Participants: Matthieu Dorier, Shadi Ibrahim, Gabriel Antoniu.

Contact: Matthieu Dorier

Presentation: Omnisc'IO is a middleware integrated in the POSIX and MPI-I/O stacks to observe, model and predict the I/O behavior of any HPC application transparently. It is based on formal grammars, implementing a modified version of the Sequitur algorithm. Omnisc'IO has been used on Grid'5000 with the CM1 atmospheric simulation, the LAMMPS molecular dynamics simulation, the GTC fusion simulation and the Nek5000 CFD simulation. Omnisc'IO was subject to a publication at SC14.

Users: Omnisc'IO is currently used only within the KerData team.

URL: http://omniscio.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.

5.2.2. Darshan-Web

Participants: Matthieu Dorier, Thomas Bouguet.

Contact: Matthieu Dorier

Presentation: Darshan-Web is a web interface for Darshan-Ruby, based on Ruby on Rails and AJAX technologies. It allows to navigate through many Darshan log files and display graphs on demand, directly on a web brother. A demo of Darshan-Web is available at http://darshan-web.irisa.fr/, which includes 2 months of logs from ANL's Intrepid supercomputer. The code of this demo is available and can be installed and used by the community.

Users: The KerData team is currently seeking potential users, in particular from Argonne National Laboratory, and will push the development further according to potential users' feedback.

URL: http://darshan-ruby.gforge.inria.fr/

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

Status: Prototype and demo available on demand.

5.2.3. JetStream

Participants: Radu Tudoran, Alexandru Costan, Gabriel Antoniu.

Contact: Alexandru Costan

Presentation: JetStream is a middleware solution for batch-based, high-performance streaming across cloud data centers. JetStream implements a set of context-aware strategies for optimizing batch-based streaming, being able to self-adapt to changing conditions. Additionally, the system provides multi-route streaming across cloud data centers for aggregating bandwidth by leveraging the network parallelism. It enables easy deployment across .Net frameworks and seamless binding with event processing engines such as StreamInsight.

Users: JetStream is currently used at Microsoft Research ATLE Munich for the management of the Azure cloud infrastructure.

License: Microsoft Public License.

Status: Prototype and demo available.

5.2.4. OverFlow

Participants: Radu Tudoran, Alexandru Costan, Gabriel Antoniu.

Contact: Alexandru Costan.

Presentation: OverFlow is a uniform data management system for scientific workflows running across geographically distributed sites, aiming to reap economic benefits from this geo-diversity. The software is environment-aware, as it monitors and models the global cloud infrastructure, offering high and predictable data handling performance for transfer cost and time, within and across sites. OverFlow proposes a set of pluggable services, grouped in a data-scientist cloud kit. They provide the applications with the possibility to monitor the underlying infrastructure, to exploit smart data compression, deduplication and geo-replication, to evaluate data management costs, to set a tradeoff between money and time, and optimize the transfer strategy accordingly.

Users: Currently, OverFlow is used for data transfers by the Microsoft Research ATLE Munich team as well as for synthetic benchmarks at the Politehnica University of Bucharest.

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

Status: Registration of the latest version with APP is in progress

5.2.5. iHadoop

Participants: Tien Dat Phan, Shadi Ibrahim.

Contact: Shadi Ibrahim

Presentation: *iHadoop* is a Hadoop simulator developed in Java on top of SimGrid to simulate the behavior of Hadoop and therefore accurately predict the performance of Hadoop in normal scenarios and under failures.

Users: iHadoop is an internal software prototype, which was initially developed to validate our idea for exploring the behavior of Hadoop under failures. iHadoop has preliminarily evaluated within our group and it has shown very high accuracy when predicating the execution time of a Map-Reduce application. We intend to integrate iHadoop within the SimGrid distribution and make it available to the SimGrid community.

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

Status: Available on Inria's forge. Registration with APP is in progress.

MESCAL Project-Team

5. New Software and Platforms

5.1. Tools to visualize and analyze traces of execution of distributed applications

Participants: Jean-Marc Vincent [correspondent], Arnaud Legrand.

The Pajé (http://paje.sourceforge.net/) generic tool provides interactive and scalable behavioral visualizations of parallel and distributed applications, helping to capture the dynamics of their executions; because of its genericity, it can be used unchanged in a large variety of contexts. Pajé Next Generation

Pajé Next Generation (https://github.com/schnorr/pajeng) is a re-implementation (in C++) and direct heir of the well-known Paje visualization tool for the analysis of execution traces (in the Paje File Format) through trace visualization (space/time view). The tool is released under the GNU General Public License 3. PajeNG comprises the libpaje library, the space-time visualization tool in pajeng and a set of auxiliary tools to manage Paje trace files (such as pj_dump and pj_validate). It was started as part of the french INFRA-SONGS ANR project. Development has continued at INF/UFRGS. Viva

Viva (https://github.com/schnorr/viva) is an open-source tool used to analyze traces (in the Paje File Format) registered during the execution of parallel or distributed applications. The tool also serves as a sandbox to the development of new visualization techniques. Current features include: Temporal integration using dynamic time-intervals Spatial aggregation through hierarchical traces Interactive Graph Visualization with a force-directed algorithm, with viva Squarified Treemap to compare processes behavior on scale, with vv_treemap

Framesoc (http://soctrace-inria.github.io/framesoc/) is the core software infrastructure of the SoC-Trace project. It provides a graphical user environment for execution-trace analysis, featuring interactive analysis views as Gantt charts or statistics views. It provides also a software library to store generic trace data, play with them, and build other analysis tools (e.g., Ocelotl). This software is developed in partnership with Nanosim. Ocelotl

Ocelotl (http://soctrace-inria.github.io/ocelotl/): Multidimensional Overviews for Huge Trace Analysis is an innovative visualization tool, which provides overviews for execution trace analysis by using a data aggregation technique. This technique enables to find anomalies in huge traces containing up to several billions of events, while keeping a fast computation time and providing a simple representation that does not overload the user.

5.2. Simulation and performance evaluation tools

Participants: Arnaud Legrand [correspondent], Luka Stanisic, Augustin Degomme, Jean-Marc Vincent, Florence Perronnin.

5.2.1. SimGrid

(see http://simgrid.gforge.inria.fr/) 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.

5.2.2. Perfect simulator

– Ψ^2 (https://gforge.inria.fr/projects/psi/) is a simulation software of markovian models. It be able to simulate discrete and continuous time models to provide a perfect sampling of the stationary distribution or directly a sampling of functional of this distribution by using *coupling from the past* The simulation kernel is based on the CFTP algorithm, and the internal simulation of transitions on the Aliasing method.

5.2.3. PEPS

– The main objective of PEPS (http://www-id.imag.fr/Logiciels/peps/) is to facilitate the solution of large discrete event systems, in situations where classical methods fail. PEPS may be applied to the modelling of computer systems, telecommunication systems, road traffic, or manufacturing systems. Development has continued at INF/UFRGS.

5.2.4. GameSeer

(http://mescal.imag.fr/membres/panayotis.mertikopoulos/publications.html) is a tool for students and researchers in game theory that uses Mathematica to generate phase portraits for normal form games under a variety of (user-customizable) evolutionary dynamics. The whole point behind GameSeer is to provide a dynamic graphical interface that allows the user to employ Mathematica's vast numerical capabilities from a simple and intuitive front-end. So, even if you've never used Mathematica before, you should be able to generate fully editable and customizable portraits quickly and painlessly.

5.3. Tools for cluster management and software development

Participant: Olivier Richard [correspondent].

The KA-Tools (http://ka-tools.imag.fr/) is a software suite developed 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.

5.3.1. KA-Deploy

(http://kadeploy3.gforge.inria.fr/) is a fast and scalable deployment system for clusters and grids. It provides a set of tools for cloning, configuring (post installation) and managing a set of nodes. Currently it can successfully deploy Linux, *BSD, Windows and Solaris on x86 and 64 bits computers. Kameleon

5.3.2. Kameleon

(http://kameleon.imag.fr/) is a simple but powerful tool to generate customized appliances. With Kameleon, you make your recipe that describes how to create step by step your own distribution. At start Kameleon is used to create custom kvm, docker, VirtualBox, ..., but as it is designed to be very generic you can probably do a lot more than that.

5.4. Infrastructure Middleware and scheduler

Participant: Olivier Richard [correspondent].

5.4.1. OAR

– The OAR project (see http://oar.imag.fr) 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). OAR 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.

5.4.2. CiGri

(http://cigri.imag.fr/) is a middleware which gathers the unused computing resource from intranet infrastructure and makes it available for the processing of 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 is s associated to the OAR resource management system (batch scheduler). Users can easily monitor and control their set of jobs through a web portal. CiGri provides mechanisms to identify job error causes, to isolate faulty components and to resubmit jobs in a safer context.

5.4.3. ComputeMode

(http://computemode.imag.fr/) is an software infrastructure that allows to extends or create a Grid through the aggregation of unused computing resources. For instance, a virtual cluster can be built using anyone's PC while not in use. Indeed, most PCs in large companies or university campus are idle at night, on weekends, and during vacations, training periods or business trips.

5.5. Platforms

5.5.1. Grid'5000

The MESCAL project-team is involved in development and management of Grid'5000 platform. The Digitalis and IDPot clusters are integrated in Grid'5000 as well as of CIMENT.

5.5.2. The ICluster-2, the IDPot and the new Digitalis Platforms

The MESCAL project-team manages a cluster computing center on the Grenoble campus. The center manages different architectures: a 48 bi-processors PC (ID-POT), and the center is involved with a cluster based on 110 bi-processors Itanium2 (ICluster-2) and another based on 34 bi-processor quad-core XEON (Digitalis) located at Inria. The three of them are integrated in the Grid'5000 grid platform.

More than 60 research projects in France have used the architectures, especially the 204 processors Icluster-2. Half of them have run typical numerical applications on this machine, the remainder has worked on middleware and new technology for cluster and grid computing. The Digitalis cluster is also meant to replace the Grimage platform in which the MOAIS project-team is very involved.

5.5.3. The Bull Machine

In the context of our collaboration with Bull the MESCAL project-team exploits a Novascale NUMA machine. The configuration is based on 8 Itanium II processors at 1.5 Ghz and 16 GB of RAM. This platform is mainly used by the Bull PhD students. This machine is also connected to the CIMENT Grid.

MOAIS Project-Team

4. New Software and Platforms

4.1. XKaapi

Participants: Thierry Gautier [correspondant], François Broquedis, Vincent Danjean, Joao Ferreira Lima.

- 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
- Additional information:

  XKaapi (http://kaapi.gforge.inria.fr, coordinator T. Gautier) is a library for high performance applications running on multi-cores/multi- processors with support for multi-GPUs. Publicly available at http://kaapi.gforge.inria.fr under CeCILL licence. XKaapi provides ABI compliant implementations of libGOMP (GCC runtime for OpenMP) and was one of the target runtime of the K'Star compiler (http://kstar.gforge.inria.fr). Direct competitors with : OMPSs (BSC), OpenMP, StarPU (Inria RUNTIME)

4.2. FlowVR

Participants: Bruno Raffin [correspondant MOAIS], Matthieu Dreher, Jérémy Jaussaud.

- ACM: D.1.3
- License: GPL and LGPL
- OS/Middelware: Unix (Linux, MacOSX, ...)
- Programming language: C/C++
- 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) is an open source middelware to augment parallel simulations running on thousands of cores with in situ processing capabilities and live steering. FlowVR offers a very flexible environment while enabling high performance asynchronous in situ and in transit processing.

  FlowVR was initially used for large scale virtual reality applications like real-time multicamera 3D modeling or telepresence. We recently retargeted FlowVR at in situ processing with development efforts focused on optimizing FlowVR performance at large scale and easing its usage in supercomputer environments.

4.3. 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.

4.4. Triva

Participant: Guillaume Huard [correspondant].

- 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.

4.5. 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.

4.6. 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.

4.7. K'Star

Participants: Thierry Gautier [correspondant], François Broquedis, Pïerrick Brunet, Philippe Virouleau, Olivier Aumage [RUNTIME project-team, Inria Bordeaux - Sud-Ouest], Samuel Thibault [RUNTIME project-team, Inria Bordeaux - Sud-Ouest], Nathalie Furmento [RUNTIME project-team, Inria Bordeaux - Sud-Ouest], Samuel Pitoiset [RUNTIME project-team, Inria Bordeaux - Sud-Ouest].

- ACM: D.1.3
- License: CeCILL
- OS/Middelware: Unix (Linux, MacOSX, ...)
- Programming language: C/C++
- Characterization of Software : A-3 / SO-2 / SM-3 / EM-2 / SDL-4
- Own Contribution: DA-4 / CD-4 / MS-4 / TPM-4
- Additional information:

  The K'Star project (http://kstar.gforge.inria.fr) supports the development of Klang, a source-to-source compiler that turns C programs with OpenMP pragmas to C programs with calls to either the StarPU or the XKaapi runtime system. K'Star is a collaboration with the EPI RUNTIME/STORM.

4.8. Kastors

Participants: Thierry Gautier [correspondant], François Broquedis, Pïerrick Brunet, Philippe Virouleau, Olivier Aumage [RUNTIME project-team, Inria Bordeaux - Sud-Ouest], Samuel Thibault [RUNTIME project-team, Inria Bordeaux - Sud-Ouest], Nathalie Furmento [RUNTIME project-team, Inria Bordeaux - Sud-Ouest].

- ACM: D.1.3
- License: CeCILL
- OS/Middelware: Unix (Linux, MacOSX, ...)
- Programming language: C/C++
- Characterization of Software : A-3 / SO-2 / SM-3 / EM-2 / SDL-4
- Own Contribution: DA-4 / CD-4 / MS-4 / TPM-4
- Additional information:

The KaStORS benchmarks suite (http://kastors.gforge.inria.fr) has been designed to evaluate the implementation of the OpenMP dependent task paradigm, introduced as part of the OpenMP 4.0 specification. KaStORS is a collaboration with the EPI RUNTIME/STORM.

4.9. Platforms

4.9.1. Multi-camera Platforms Grimage and Kinovis

MOAIS has managed with the LJK-Inria Morpheo team the Grimage platfrom (http://grimage.inrialpes.fr) dedicated to off-line and on-line 3D modeling from multiple cameras and telepresence. In 2012, we received an Equipex funding, Kinovis (http://kinovis.inrialpes.fr), to renew this platform. Kinovis will be operational by early 2015 and will consist of 68 cameras, a compute cluster and a large acquisition space. FlowVR is the software backbone of both platforms for live processing. MOAIS is participating to the FP7 infrastructure project Visionair to enable European research teams to experiment on both platforms.

4.9.2. HPC Platforms Grid'5000 and Ciment

MOAIS is involved in the national platform Grid'5000, the regional mezzo center Ciment and obtained in 2014 with the Mescal and Erods team a grant (FAIRE from Grenoble-INP and LIG) to buy various large NUMA nodes and accelerators that will be integrated into the Grid'5000 infrastructure.

ROMA Team

5. New Software and Platforms

5.1. MUMPS

Participants: Patrick Amestoy, Alfredo Buttari, Jean-Yves L'Excellent [correspondent], Chiara Puglisi, Wissam M. Sid-Lakhdar, Bora Uçar.

MUMPS (for *MUltifrontal Massively Parallel Solver*) see http://mumps-solver.org is a software package for the solution of large sparse systems of linear equations. It implements a direct method, the so called multifrontal method; it is a parallel code capable of exploiting distributed-memory computers as well as multithreaded libraries; its main originalities are its numerical robustness (including partial threshold pivoting in distributed-memory environment) and its wide range of features.

The latest public release is MUMPS 4.10.0 (May 2011); the new release is scheduled for February 2015 and will be under the Cecill-C licence, following an agreement between CERFACS, CNRS, ENS Lyon, INPT, Inria and University of Bordeaux.

RUNTIME Team

5. New Software and Platforms

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] [20], 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 30 000 lines of C.
- http://www.open-mpi.org/projects/hwloc

5.3. Network Locality

Participant: Brice Goglin.

- *Netloc Locality* (NETLOC) is a library that extends hwloc to network topology information by assembling hwloc knowledge of server internals within graphs of inter-node fabrics such as Ethernet or Infiniband.
- NETLOC targets the same challenges as hwloc but focuses on a wider spectrum by enabling cluster-wide solutions such process placement [21].
- NETLOC is developed in collaboration with University of Wisconsin in LaCrosse and Cisco, within the OPEN MPI project.
- NETLOC is composed of 15 000 lines of C. It was recently merged in the HWLOC repository was better integration.
- http://netloc.org

5.4. 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 8 000 lines of C. Its main contributor is Brice GOGLIN.
- http://runtime.bordeaux.inria.fr/knem/

5.5. 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 [45] or the recent iWarp [44] 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.gforge.inria.fr/

5.6. StarPU

Participants: Olivier Aumage, Andra Hugo, Nathalie Furmento, Raymond Namyst, Marc Sergent, Samuel Thibault, Pierre-André Wacrenier.

- STARPU permits high performance libraries or compiler environments to exploit heterogeneous multicore machines possibly equipped with GPGPUs or Xeon Phi 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 *scheduling contexts* which can be used to partition computing resources. Scheduling contexts can be dynamically resized to optimize the allocation of computing resources among concurrently running libraries.
- STARPU provides integration in MPI clusters through a lightweight DSM over MPI.
- STARPU provides a scheduling platform, which makes it easy to implement and experiment with scheduling heuristics
- 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.
- STARPU has support for simulating the execution, by using the simgrid simulator, which allows to reproduce experiments on a remote system, or to even virtually modify the platform used to run the application
- STARPU has been extended to provide runtime support for the KLANG-OMP OpenMP compiler. StarPU's OpenMP runtime support is compliant with most OpenMP 3.0 constructs, and also supports new dependent tasks and accelerated targets constructs introduced by OpenMP 4.0.
- http://runtime.bordeaux.inria.fr/StarPU/

5.7. Klang-OMP

Participants: Olivier Aumage, Nathalie Furmento, Samuel Pitoiset, Samuel Thibault.

- The KLANG-OMP software is a source-to-source OpenMP compiler for languages C and C++. It is developed as part of the Inria development action "ADT K'STAR " jointly managed by Inria teams MOAIS (Inria Montbonnot) and RUNTIME (Inria Bordeaux - Sud-Ouest). The KLANG-OMP compiler translates OpenMP directives and constructs into API calls from the StarPU runtime system or the XKaapi runtime system (XKaapi is developed by the MOAIS team).
- The KLANG-OMP compiler is virtually fully compliant with OpenMP 3.0 constructs.
- The KLANG-OMP compiler supports OpenMP 4.0 dependent tasks and accelerated targets.

5.8. Kastors

Participants: Olivier Aumage, Nathalie Furmento, Samuel Pitoiset, Samuel Thibault.

- The KASTORS software is a suite of benchmarks for testing the performance of OpenMP compilers on codes making use of the new dependent tasks OpenMP 4.0 constructs. It is ported and maintained as part of the Inria development action "ADT K'STAR " jointly managed by Inria teams MOAIS (Inria Montbonnot) and RUNTIME (Inria Bordeaux - Sud-Ouest). It is constituted of well known computing kernels that have been ported on the OpenMP 4.0 dependent tasks model. The KASTORS suite has been introduced to the OpenMP community during the IWOMP 2014 conference [32].

5.9. NewMadeleine and PIOMan

Participant: Alexandre Denis.

- NEWMADELEINE is a 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.
- PIOMAN is a generic framework to be used by communication libraries, that brings seamless asynchronous progression of communication by opportunistically using available cores. It uses system threads and thus is composable with any runtime system used for multithreading.
- PIOMAN is closely integrated with the NEWMADELEINE communication library and PadicoTM.
- The reference software development branch of the NEWMADELEINE software consists in 60 000 lines of code. NEWMADELEINE is available on various networking technologies: Myrinet, Infiniband, Quadrics and ETHERNET. It is developed and maintained by Alexandre DENIS.
- http://pm2.gforge.inria.fr/newmadeleine/

5.10. 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://pm2.gforge.inria.fr/PadicoTM/

5.11. MAQAO

Participants: Denis Barthou, Olivier Aumage, Christopher Haine, James Tombi A Mba.

- 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 for Intel x86 and ARM architectures. For x86 architecture, it can insert 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 and ARM architectures. This model analyzes performance of the codes on the architectures and provides some feed-back hints on how to improve these codes, in particular for vector 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 European FP7 project "MontBlanc".

5.12. QIRAL

Participants: Denis Barthou, Olivier Aumage.

- 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.

5.13. 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.
- Some core algorithms are parallel to speed-up the execution.
- 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.

TYREX Project-Team

5. New Software and Platforms

5.1. XML Reasoning Solver

Participants: Pierre Genevès, Nabil Layaïda, Nils Gesbert, Louis Jachiet, Nicola Guido.

The XML Reasoning Solver is a tool for the static analysis of queries and schemas based on our theoretical advances [9]. It allows automated verification of properties that are expressed as logical formulas over trees. A logical formula may for instance express structural constraints or navigation properties (like e.g. path existence and node selection) in finite trees.

The reasoner is built on top of a finite tree logic solver for a new modal logic equipped with recursion and backward axes. The solver is very fast in practice and uses symbolic techniques (Binary Decision Diagrams). The solver has been recently extended to support functions, parametric functions and polymorphic subtyping. One notable difficulty was to elaborate many advanced optimizations with symbolic implementation techniques. The logical solver significantly advances the state of the art. In particular, it is the first implementation that effectively solves the query containment problem for a large fragment of the XPath query language. It supports all navigation axes and regular tree constraints. Although researchers had studied XPath satisfiability before, such prior works were either unimplementable or deemed to explode even for tiny examples. As of 2014, it is still the only implementation actually capable of solving this problem in practice for real world instances.

The reasoner includes compilers and various static analyzers for web query and schema languages. This includes compilers for XPath, for XML schemas (DTDs, XML Schemas, Relax NGs) into logical formulas, parsers, benchmarks, and libraries for automated testing. Various difficulties reside in the compilation of real-world queries, including compiling XPath queries into fixed-point logics, developing specific implementation techniques in order to avoid worst case blow-ups as much as possible when e.g. supporting unordered XML attributes among (ordered) XML elements, etc. The reasoner also generates counter-examples that allow program defects to be reproduced independently from the analyzer.

The off-line version of the solver (with a native library) is fast and up-to-date with the latest advances. We developed and deployed an interactive web interface to make the solver available to the international scientific community. For this purpose, we redesigned the libraries used for the manipulation of binary decision diagrams (BDDs) so that they could used in a fully concurrent and multithreaded manner. This is in order to allow several instances of the logical solver to run concurrently for several users on a web server (GWT-based), while decreasing performance as less as possible.

The reasoner helps us to guide and validate our approach. We continue to develop, maintain and use it on an almost-daily basis.

5.2. XQuery type-checker

Participants: Pierre Genevès, Nabil Layaïda, Nils Gesbert.

This prototype implements a sound static type-system for XQuery, which, as of december 2014, is the most precise type system known for XQuery. It supports the static typing of backward axes that no other does nor is supported in the XQuery recommendation. It also includes precise typing for conditional statements which is challenging as such statement are usually sensitive to the program context. Our type checker successfully verifies complex programs for which existing type-checkers (either known from the literature or those developed in commercial software) fail by reporting false alarms. One major benefit is to allow the cost of validation to be deferred from runtime to compile-time (once only). This prototype is implemented in Scala and interacts with the solver by issuing externals calls for deciding complex subtyping relations. This prototype is described in preprint [20]

5.3. CSS Analyzer

Participants: Pierre Genevès, Nabil Layaïda, Marti Bosch Padros.

This software now consists in two distinct prototypes: two static analyzers (with a different purpose) that share a common compiler for CSS. The first prototype is used for bug detection and verification of a cascading style sheet (CSS) file. It involves a compiler for CSS rules (and in particular selectors) into logical formulas, adapted for the semantics of CSS (see the initial WWW'12 paper). The second prototype performs automated refactoring for size reduction of CSS style sheets. It reuses the first compiler and the logical solver for detecting which rules can be refactored and how. It implements various optimisation techniques (like early pruning), for the purpose of dealing with large-size real CSS files. This prototype reduces the size of CSS files found in the most popular websites (such as CNN, facebook, Google Sites, Apple, etc.) by up to 30% while preserving their semantics [13].

5.4. ClaireCourseMaker Library

Participants: Nicolas Hairon, Cécile Roisin, Nabil Layaïda.

The goal of the ClaireCourseMaker is to provide direct and visual editing tools for structuring, annotating and timeline-based authoring of continuous content such as audio or video. I it is mainly devoted to the synchronisation and layout of pedagogical material (video, slides, chaptering, etc.) and enables the incorporation of rich media content in MOOCs. The underlying technology is based on Web standards and relies on the open source JavaScript Popcorn library and Popcorn Maker web application developed by the Mozilla Foundation. The tool is a wysiwyg web-based authoring tool which benefits from the generic features of Popcorn and offers structuring methods such chaptering and container-based synchronisation.

ClaireCourseMaker is the direct follow-up tool of the Timesheet library developed in the project. Timesheet library is a cross-browser JavaScript implementation for scheduling the dynamic behaviour of HTML5 content. It uses and provides a reference implementation for declarative synchronisation markup such as SMIL Timing and Synchronization and SMIL Timesheets.

ClaireCourseMaker is developed in collaboration with the OpenClassrooms company in the context of the Claire project (see section 7.1.1).

5.5. Interactive eXtensible Engine (IXE)

Participants: Nabil Layaïda, Pierre Genevès, Thibaud Michel, Mathieu Razafimahazo.

PDRTrack is a localization utility running on iOS or Android smartphones used for recording and playing data sets (accelerometer, gyroscope, barometer and magnetometer values) to study the effect of different pedometer and map matching parameters on indoor and outdoor localization accuracy. This application uses the PDR library, written in C++, which provides the user's location in real time based on the interpretation of mobile phone sensors. Three main modules have been designed to build this localization system:

- a pedometer that estimates the distance the user has walked and his speed
- a motion manager that enables data set recording and simulation but also the creation of virtual sensors or filters (e.g gyroscope drift compensation, linear acceleration, altimeter)
- a map-matching algorithm that provides location estimates on a given OpenStreetMap description and the current user's trajectory

The PDR library is a central component of the VENTURI project. It has been used for applications such guiding a visually impaired people. Others partners have used this localisation system for retrieving a scale factor needed for the computer vision part (i.e SLAM).

GPS navigation systems, when used in an urban environment, are limited in precision and can only give instructions at the level of the street and not of the pavement or corridor. GPS is also limited to outdoor navigation and requires some transitioning system when switching to indoor navigation.

PDRTrack is embedded in IXE. IXE is an urban pedestrian navigation system based on Inertial Measurement Units (IMU) and running on mobile phones with onboard geographic data and a routing engine. IXE allows augmented reality queries on customised embedded geographical data. Queries on route nodes or POIs, on ways and relations are predefined for efficiency and quality of information. Following a web paradigm, IXE can be seen as web browser for XML documents describing navigation networks. by using the micro-format concept, one can define inside OpenStreetMap a complex format for pedestrian navigation networks allowing navigation at the level of pavements or corridors.

The big advantage of IXE is that it relies on a standard OpenStreetMap editor called JOSM to create navigation networks and augmented reality content. IXE browser reads OSM documents and produces from them visible or audible navigation information. IXE is composed of three engines, one for dead-reckoning navigation, one for interactive audio and the last one for Augmented Reality visual information.

ASCOLA Project-Team

5. New Software and Platforms

5.1. btrCloud (and Entropy)

Participants: Jean-Marc Menaud [correspondent], Guillaume Le Louët, 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 centers. 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, on the long run, or in order to predict future trends. This feature includes boards for performance evaluation and analysis as well as trends estimation.

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. The latter provides optimized quality of service properties for applications that are hosted in the virtualized datacenters.

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

5.2. EScala and JEScala

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

AOP, inheritance, event-based programming, events, declarative events, asynchronous events, join operator, Scala

EScala is an extension of the programming language Scala with support for events as object members. EScala combines ideas of event-driven, aspect-oriented and functional reactive programming.

Events are natural abstractions for describing interactive behavior as part of an object interface. In conventional object-oriented languages, events are implemented indirectly, typically using the Observer pattern. C# eliminates the corresponding glue code and directly supports events as object members. However, events are still *explicitly* triggered at specific locations within the program.

EScala goes much further. First, it also supports *implicit* events. Akin to join points in aspect-oriented languages, these events are implicitly produced at specific execution points, such as the beginning or the end of the execution of a method. Second, *declarative events* make it possible to compose events using logical operators as well as to filter them and alter their content.

EScala events are fully integrated with object-oriented features. An event is defined in the context of its owner object. Event definitions are inherited in subclasses and event uses are late-bound. Unlike typical aspect-oriented languages, EScala preserves object-oriented encapsulation and modular reasoning.

JEScala extends EScala with support for concurrent programming (see Sec. 6.2). Events can be declared as *asynchronous* so that their handling takes place concurrently. A new composition operator, the *join* operator, inspired by the join calculus, can also be used to synchronize the concurrent activities created by asynchronous events and communicate between them.

This is joint work with the Software Technology Group at TU Darmstadt.

Prototype implementations of these languages are available through http://www.stg.tu-darmstadt.de/research.

5.3. CSLA

Participants: Thomas Ledoux [correspondent], Yousri Kouki.

Service-level agreement, Cloud computing, elasticity

Verifying non-functional properties like performance, dependability, energy consumption and economical costs of Cloud is challenging today due to ad hoc management in terms of Quality-of-Service (QoS). We believe that a differentiating element between Cloud computing environments will be the QoS and the Service-Level Agreement (SLA) provided by the Cloud.

CSLA, the Cloud Service Level Agreement language, allows the definition of SLA properties for arbitrary Cloud services (XaaS). CSLA addresses QoS uncertainty in unpredictable and dynamic environment and provides a cost model of Cloud computing. Besides the standard formal definition of contracts – comprising validity, parties, services definition and guarantees/violations – CSLA is enriched with features, such as QoS degradation and an advanced penalty model, thus introducing fine-grained language support for Cloud elasticity management [27][26].

CSLA is available at http://www.emn.fr/z-info/csla.

5.4. SAdapt

Participants: Ronan-Alexandre Cherrueau [correspondent], Mario Südholt.

Service-oriented systems, distributed programming, event-based programming, workflow patterns

The SAdapt tool provides an implementation of workflow adaptation patterns and allows the transformation of service-oriented systems implemented using Apache's CXF service infrastructure in terms of high-level declarative service transformations. The transformations are defined using an expressive language that supports matching of the execution of service-based systems in terms of flexible patterns over service compositions.

The SAdapt tool has partially been developed and is employed in the A4Cloud EU project (see Sec. 8.3) as a basis for our work on the enforcement of accountability properties in complex cloud-based systems.

The SAdapt tool and its application, notably to the security hardening of service systems that use OAuth 2 for the authorization of resource accesses is available at http://a4cloud.gforge.inria.fr/doku.php?id=start:advservcomp.

In 2014, we have used and extended the tool in order to investigate accountability properties of service-based applications, see Sec. 6.3 .

5.5. SimGrid/VMPlaces

Participants: Takahiro Hirofuchi, Adrien Lebre [correspondent], Jonathan Pastor, Flavien Quesnel, Mario Südholt.

Simulation, Virtualization, Cloud computing, VM placement

SimGrid is a toolkit for the simulation of algorithms executed on large-scale distributed systems. Developed for more than a decade, it has been used in a large number of studies described in more than 100 publications. In 2013, ASCOLA with the support of the SimGrid core-developers, designed and implemented additional capabilities, in particular the Virtual Machine abstraction, enabling to address Cloud Computing related concerns.

Developed, first, in an experimental repository, the integration of these extensions into the master branch of SimGrid has been achieved during Summer 2014. The principal role of ASCOLA is now to ensure the maintenance of this portion of the code with respect to the evolutions of the SimGrid toolkit (such as for instance the recent port of the SimGrid kernel in C++).

Although the virtualization extensions are recent, several projects leveraging them have been already proposed. [0] Among them, ASCOLA is working on dedicated framework to evaluate and compare VM placement algorithms. Entitled VMPlaces, this framework is composed of two major components: the injector and the VM placement algorithm. The injector is the generic part of the framework (i.e. the one you can directly use) while the VM placement algorithm is the part you want to study (or compare with available algorithms). Currently, the VMPlaceS is released with three algorithms:

- Entropy, a centralized approach using a constraint programming approach to solve the placement/reconfiguration VM problem
- Snooze, a hierarchical approach where each manager of a group invokes Entropy to solve the placement/reconfiguration VM problem. Note that in the original implementation of Snooze, it is using a specific heuristic to solve the placement/reconfiguration VM problem. As the sake of simplicity, we have simply reused the entropy scheduling code.
- DVMS, a distributed approach that dynamically partitions the system and invokes Entropy on each partition.

SimGrid is available at http://simgrid.gforge.inria.fr.
VMPlaces is available at http://beyondtheclouds.github.io/VMPlaceS/

[0] The list of the projects is available at : http://simgrid.gforge.inria.fr/contrib/clouds-sg-doc.html

DIVERSE Project-Team

5. New Software and Platforms

5.1. Kermeta

Participants: Benoit Combemale [correspondant], Olivier Barais, Arnaud Blouin, Didier Vojtisek, Benoit Baudry, Thomas Degueule, David Mendez, Erwan Bousse, Francois Tanguy, Fabien Coulon.

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 Ecore industrial standard and aligned with the EMOF 2.0 OMG standard, 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.

Kermeta 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.

In 2014, we have continued the refactoring of Kermeta to leverage on Xtend. The Kermeta action language is now defined as an extension of Xtend, by proposing model-specific features (e.g., model type, containment, opposite) and an open class mechanism for aspect weaving. The main objective of this new refactoring was to benefit from the non model–specific features of Xtend (including the basics of the action language and its respective tooling such as editor, type checker and compiler), and to focus in our development on innovative solutions for MDE.

More precisely, in addition to an Xtend extension dedicated to model manipulation, we started to integrate in Kermeta various facilities to support software language engineering (slicing, pruning, reuse, variability management, etc). In 2014, we improved this software language engineering feature (currently named k3sle/Melange) in order to offer a functional model typing system allowing safe model polymorphism. This system enables reuse of algorithms and transformations accross different metamodels, as well as language inheritance, evolution and interoperability.

Moreover, while this version of Kermeta is a DSML development workbench that provides good support for developping independent DSMLs, little or no support is provided for integrated use of multiple DSMLs. The lack of support for explicitly relating concepts expressed in different DSMLs makes it very difficult for developers to reason about information spread across models describing different system aspects.

According to Google Scholar [0], the Kermeta platform was used or cited in more than 1300 papers.

5.2. FAMILIAR

Participants: Mathieu Acher [correspondant], Olivier Barais, Guillaume Bécan, Aymeric Hervieu, Julien Richard-Foy, Sana Ben Nasr, Edward Mauricio Alferez Salinas, João Ferreira Filho, Didier Vojtisek, Benoit Baudry.

[0] http://scholar.google.fr/scholar?q=kermeta+model

Modeling and reasoning about configuration options is crucial for the effective management of configurable software systems and product lines. The FAMILIAR project provides dedicated languages, APIs, and comprehensive environments for that purpose. Specifically, FAMILIAR provides support for feature models (by far the most popular notation). The feature models formalism has been studied for more than 20 years [98], and it is widely used in the industry [100]. FAMILIAR (for FeAture Model scrIpt Language for manIpulation and Automatic Reasoning) provides a scripting language for importing, exporting, composing, decomposing, editing, configuring, computing "diffs", refactoring, reverse engineering, testing, and reasoning about (multiple) feature models. For interoperability, many bridges with existing feature modeling languages are implemented. All these operations can be combined to perform complex variability management tasks: extraction of feature models from software artifacts [87], product line evolution [89], management of multiple models [88] [75], [76], model-based validation of SPLs [22], large scale configuration of feature models [122], etc. The level of maturity of the FAMILIAR platform is TRL 3 (*i.e.* New technology tested Prototype built and functionality demonstrated through testing over a limited range of operating conditions. These tests can be done on a scaled version if scalable).

Main innovative features:

- reverse engineering of variability models from multiple kinds of artefacts;
- composition of multiple variability models (e.g., for combining different sources of variability);
- slicing of variability model (e.g., for scheduling a configuration process in different steps);
- connection with the Common Variability Language (CVL);
- support of advanced variability constructs (e.g., attributes, multi-features, meta-information);
- Web-based, comprehensive environment (WebFML [42]).

Impact:

The results are connected to the CVL standardization initiative. From a research perspective, FAMILIAR helps to support all the research activity on variability modeling (e.g., design of new operators, benchmarking). Several tutorials and tool demonstrations [42], [25] have been performed at SPLC (the major conference in software product lines), at ECOOP, at CIEL and MODELS in 2012 and 2013. FAMILIAR is also used in the context of teaching activities. From an industrial perspective, the languages and tools have already been applied in practical contexts in different application domains (medical imaging, video surveillance, system engineering, web configurators, etc.) and for various purposes. This platform is also used for supporting industrial transfer activity with companies such as Thales. FAMILIAR is involved in several research projects (e.g., in the Merge ITEA project, in the MOTIV project, in the VaryMDE project).

FAMILIAR is distributed under the terms of the LGPL and EPL open source license.

See also the web page familiar-project.github.com.

- Version: 1.3
- Programming language: Java, Scala

5.3. Kevoree

Participants: Olivier Barais [correspondant], Johan Bourcier, Noel Plouzeau, Benoit Baudry, Maxime Tricoire, Jacky Bourgeois, Inti G. Herrera, Ivan Paez, Francisco Acosta, Mohamed Boussaa.

Kevoree is an open-source models@runtime platform [0] to properly support the dynamic adaptation of distributed systems. Models@runtime basically pushes the idea of reflection [132] 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.

[0] http://www.kevoree.org

Kevoree has been influenced by previous work that we carried out in the DiVA project [132] and the Entimid project [135]. With Kevoree we push our vision of models@runtime [131] 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 [0].
- SpringSource Dynamic Module [0]
- GCM-Proactive [0]
- OSGi [0]
- Chef [0]
- Vagran [0]

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:

A tutorial have been performed at the Comparch conference in July 2014 and at the Middleware conference in december 2014. See also the web page http://www.kevoree.org.

In 2014, we mainly created a new implementation in JavaScript and we created an implementation for system containers for driving resources using Kevoree. We also use Kevoree in the context of Mohammed's PhD to create testing infrastructure on-demand.

- Version: 5.1.4
- Programming language: Java, Scala, Kermeta, Kotlin, Javascript

[0] http://frascati.ow2.org
[0] http://spring.io/
[0] http://proactive.inria.fr/
[0] http://www.osgi.org
[0] http://wiki.opscode.com/display/chef/Deploy+Resource
[0] http://vagrantup.com/

FOCUS Project-Team

5. New Software and Platforms

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.

During 2014, the development of Jolie 1.1 has been completed (the release is due for the first half of January 2015). It is the result of about 600 commits, including more than 30 new standard library APIs, 100 bugfixes, and 100 improvements to the Jolie interpreter and libraries. Highlights:

- A new hierarchical semantics for handling sub-programs loaded in higher-order Jolie services.
- Support for "abstract locations". This enables the writing of extensions that automatically fetch the bindings to the external services needed by a Jolie program. We plan to use this feature to develop binding procedures that ensure correctness.
- Introduction of a tracer option for the Jolie interpreter, which displays the execution trace of a Jolie program (useful for debugging).
- Substantial improvements to memory management of higher-order programs.
- Improved integration with web applications, which supports new techniques for handling the evolution of legacy web applications using the composition primitives of Jolie.

Moreover Jolie Enterprise has been released: this is an administrative tool that allows one to deploy Jolie services on remote nodes. Jolie Enterprise is able to manage services that run on different nodes on different machines, tracking all messages exchanged between services and viewing the log on GUI so that one can have a report of what happened in the system. Currently there are about 15 installations of Jolie Enterprise at SME in clothing, construction and manufacturing.

5.2. Others

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

- *Deadlock analysis* (http://df4abs.nws.cs.unibo.it/).

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

Reversible debugging provides developers with a way to execute their applications both forward and backward, seeking the cause of an unexpected or undesired event. We have developed CaReDeb, the first prototype of a causal-consistent reversible debugger. Causal consistent here means that independent actions are undone independently, while dependent actions are undone in reverse order. This allows the programmer to concentrate on the threads responsible of the bug, independently of the actual interleaving. CaReDeb provides primitives that given a misbehaviour, e.g., a variable has not the expected value, allow one to go back to the action responsible for it, e.g., the one that assigned the wrong value to the variable. Notably, the programmer has no need to know which thread the action belongs to, since this is found automatically by the debugger. The procedure can be iterated till the bug is found. CaReDeb targets a fragment of the language Oz, which is at the basis of Mozart. The considered fragment provides functional variables, procedures, threads, and asynchronous communication via ports.

- *AIOCJ* (http://www.cs.unibo.it/projects/jolie/aiocj.html).

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

  As partners of the Aeolus project we have developed a tool for the automatic synthesis of deployment plans. A deployment plan is a sequence of actions that, when performed, allows the deployment of a given configuration of components. METIS (Modern Engineered Tool for Installing Software systems) is a tool that enables one to automatically generate a deployment plan, starting from a description of the configuration following the Aeolus model. The software is open source. It is written entirely in OCaml and is about 3.5K lines of source code. The tool is based on theoretical results that guarantee its soundness and completeness, while maintaining polynomial computational complexity. METIS already showed its effectiveness in practice by handling synthesized problem instances with hundreds of components in less than a minute. We are currently validating Metis in a production environment by integrating it in Armonic, an infrastructure for cloud application deployment in OpenStack cloud systems developed by the Mandriva company.
- *SUNNY-CP* (https://github.com/jacopoMauro/sunny-cp)

  Within the Constraint Programming (CP) paradigm, a portfolio approach enables to combine a number of different constraint solvers in order to create a globally better solver, dubbed a portfolio solver. After several empirical evaluations (e.g., [22], [23], [24]) we have decided to develop *SUNNY-CP*, a portfolio solver for solving both Constraint Satisfaction Problems and Constraint Optimization Problems. The goal of *SUNNY-CP* is to provide a flexible, configurable, and usable CP portfolio solver that can be set up and executed just like a regular individual CP solver. To the best of our knowledge, *SUNNY-CP* is the only sequential portfolio solver able to solve generic CP problems, and it was the only portfolio solver that attended the MiniZinc Challenge 2014 (i.e., the only active international competition to evaluate the performance of CP solvers). (*SUNNY-CP* performed very well, ranking 4th in the competition and receiving an honourable mention by the challenge organizers.) The application of *SUNNY-CP* in the optimization problems defined within the Aeolus project have lead to time improvements beyond an order of magnitude. *SUNNY-CP* is mainly written in Python, and we are currently enhancing the tool in order to make it more usable, flexible, and parallel (i.e., able to properly exploit multiple cores).

The sofware below have not undergone substantial modifications during 2014.

- *Croll-pi Interpreter* (http://proton.inrialpes.fr/~mlienhar/croll-pi/implem/). 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 have developed an interpreter for croll-pi using Maude.
- *IntML* is a functional programming language guaranteeing sublinear space bounds for all programs [51]. See the Activity Reports of previous years (in particular 2010) for more details.
- *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. See the Activity Reports of previous years (in particular 2010) for more details.

INDES Project-Team

5. New Software and Platforms

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 software packages can be downloaded from the INDES web site:

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

5.2. Language-based Security

Participants: José Fragoso Santos, Tamara Rezk [correspondant].

5.2.1. JavaScript Library iflowtypes.js

The JavaScript library iflowtypes.js is designed to type secure information flow in JavaScript. iflowtypes.js has two main modes of operation: fully static and hybrid. In the hybrid mode, the program to be typed is instrumented with runtime assertions that are verified at runtime. By deferring rejection to runtime, the hybrid type system is able to type more programs than fully static mechanisms. This library is available at the URL: http://j3fsantos.github.io/PersonalPage/TypeSystem/.

5.2.2. JavaScript Library iflowsigs.js

The JavaScript library iflowsigs.js is designed to inline an information flow monitor into JavaScript code. iflowsigs.js supports is able to track information flow even in programs that interact with arbitrary Web APIs. This library is available at the URL: http://j3fsantos.github.io/PersonalPage/IFMonitor/.

5.3. Web programming

Participants: Yoann Couillec, Vincent Prunet, 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. It is released under the GPL license. It is available at http://hop.inria.fr.

5.4. Old software

5.4.1. 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.

5.4.2. 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.

5.4.3. 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.

5.4.4. 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.

5.4.5. The Bigloo compiler

The programming environment for the Bigloo compiler [7] 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.

5.4.6. 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.

5.4.7. 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.

5.4.8. 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/.

5.4.9. IFJS compiler

The IFJS compiler is applied to JavaScript code. The compiler generates JavaScript code instrumented with checks to secure code. The compiler takes into account special features of JavaScript such as implicit type coercions and programs that actively try to bypass the inlined enforcement mechanisms. The compiler guarantees that third-party programs cannot (1) access the compiler internal state by randomizing the names of the resources through which it is accessed and (2) change the behaviour of native functions that are used by the enforcement mechanisms inlined in the compiled code.

The compiler is written in JavaScript and can be found at http://www-sop.inria.fr/indes/ifJS.

PHOENIX Project-Team

5. New Software and Platforms

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

Participants: Charles Consel [correspondent], Milan Kabac, Paul Van Der Walt, Adrien Carteron, Alexandre Spriet.

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 [35]. This layer is dedicated to an architectural pattern commonly used in the pervasive computing domain [31]. 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 2 .

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

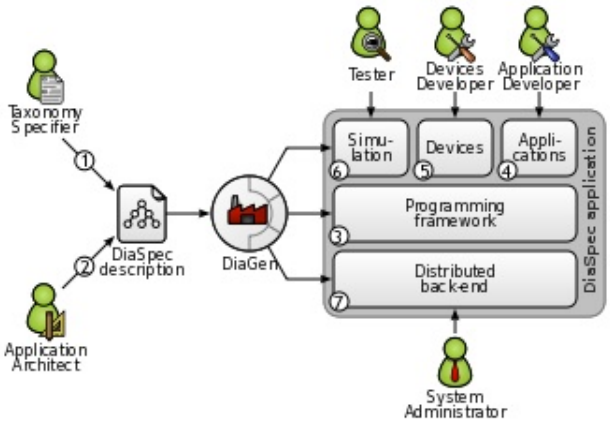

Figure 2. DIASUITE *Development Cycle*

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 [31]. 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

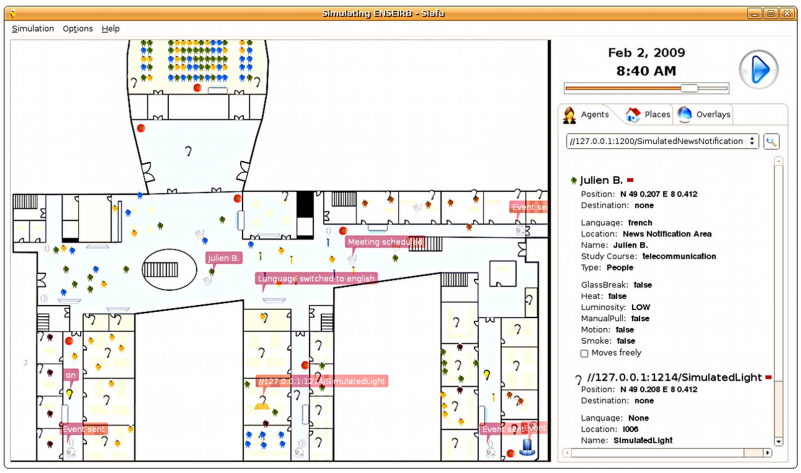

Figure 3. A screenshot of the DIASIM *simulator*

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 3 .

5.2. DiaSuiteBox: an Open Orchestration Platform

Participants: Charles Consel, Adrien Carteron, Alexandre Spriet, Milan Kabac.

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.

5.2.1. DiaSuiteBox platform architecture

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 the behavior of its applications are innocuous and correct with respect to the provided information. Finally, DiaSuiteBox provides an extensible software architecture. This allows the easily connect new device technologies to the platform. For example, the support for new wireless communication technologies such as Zigbee, Z-Wave or Sigfox can be easily added to the DiaSuiteBox platform.

More details can be found on the web page http://diasuitebox.inria.fr.

The iQSpot startup uses DiaSuiteBox as a software platform to ease the management of Smart Buildings. In this project, the DiaSuiteBox platform is first used to host building management functionalities such as lighting management, heating/ventilating/air conditioning management, energy efficiency monitoring. It is also used to host software drivers that allow the building management functionalities to interact with the connected devices deployed in buildings. These devices can use wired communication technologies such LonWorks, BACNet or KNX, as well as wireless communication technologies such as Z-Wave or Zigbee.

5.3. School+ Apps: Assistive tablet applications for school Inclusion

Participants: Charles Consel [correspondent], Hélène Sauzéon, Charles Fage, Cécile Magnier.

School+ is a package of 7 applications. Three applications are assistive applications, guiding the child doing specific tasks. Three others are training applications made as serious games, addressing specific skills. The last application is a meta-application, comprising a link to the three training applications, with an access to statistics of their usage. For each application, data are separated from the design, meaning that every element of each application (pictures, texts, settings, etc.) can be changed at any time. Each application records a log file containing all the interactions performed by the child.

5.3.1. Assistive applications:

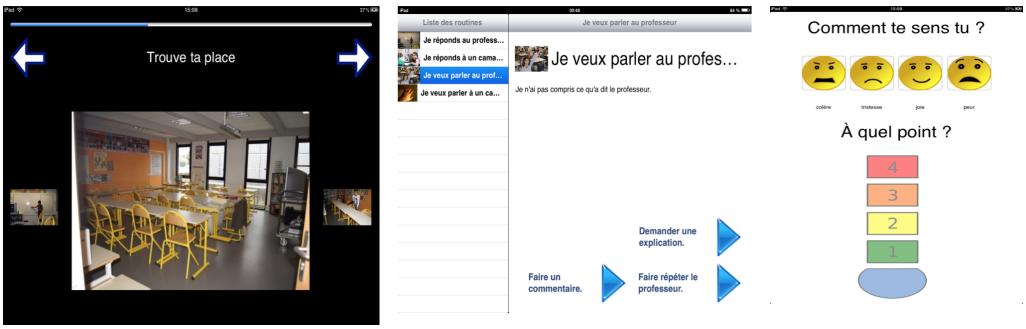

Figure 4. Assistive applications

- *Routines application*. This application shows a list of tasks, with a short description. After clicking the starting button, a specific slideshow is shown; it decomposes a task into steps. For each step, a text and a picture can be displayed. Thumbnail of previous and next steps are also displayed. This application guides the child through classroom situations: entering classroom, taking school materials out of a backpack, writing notes, handling agenda, leaving the classroom.
- *Communication application*. With the same design, the assistance provided by this application targets to communicating situations inside the classroom. The application covers four scenarios addressing two interaction situations (initiating and answering the interaction) and two types of interlocutors (professor and classmate). For each scenario, different slideshows guide the child, depending on the goal of the interaction.
- *Emotion Regulation application*. This application aims to assist the child to self-regulate his/her emotions. Four simplified emoticons are proposed to the child to choose from: anger, sadness, joy and fear. Then, (s)he selects a level of intensity via a thermometer with a scale from 1 to 4. In response, the application delivers different multimedia contents according to the level selected to help the child regulate his/her emotions. Typically, a text (breathing instructions) are shown at level 1, pictures at level 2, a video at level 3 and another text at level 4.

5.3.2. Training applications:

These three applications are serious games with increasing levels of difficulties, reachable after a ratio of good answers has been attained.

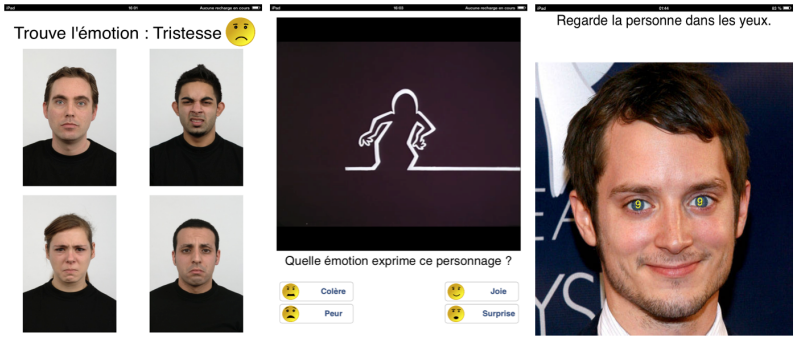

Figure 5. Training applications

- *Emotion Recognition application: pictures*. In this application, the child is instructed to identify a specific emotion among 4 pictures showing different people exhibiting an emotion. Seven emotions are involved in this application: joy, sadness, fear, anger, surprise, disgust and neutral. The emotion to be recognized is displayed together with its simplified emoticon. The type of pictures changes with the difficulty level: level 1 contains pictures of unfamiliar people and level 2 contains pictures of friends and relatives of the child.
- *Emotion Recognition application: videos*. In this application, the child is presented with a fragment of an animated cartoon. At some point, the video stops and the child is asked to identify the emotion of the character. Four emotions are involved in this application: joy, sadness, fear and anger. Videos are slowed down, with a speed percentage that can be changed at each level. Videos change with difficulty level: level 1 contains videos of a very basic cartoon (only one cartoon character drawn by basic form un-textured), level 2 contains a video of more sophisticated cartoons and level 3 contains movies with actors.
- *Attention Training*. In this application, the child is presented a picture of a face and asked to make eye contact with it. Second, a symbol appears briefly in the eyes of the character. Third, the child is asked to identify the symbol shown in the previously displayed picture, to make sure he kept eye contact. The speed at which the symbol appears and disappears is changed according to the difficulty level. Types of pictures also change with the level : level 1 contains pictures of faces and level 2 contains pictures of classroom situations.

5.4. HomeAssist: A Platform for Assistive Living

Participants: Charles Consel, Loïc Caroux [correspondent], Thomas Freslon, Adrien Carteron, David Daney, Lucile Dupuy, Geoffrey Escojido, Bernard N'Kaoua, Hélène Sauzéon, Alexandre Spriet.

The HomeAssist platform proposes a systemic approach to introducing an assistive technological platform for older people. To do so, we formed a trans-disciplinary team that allows (1) to identify the user needs from a gerontological and psychological viewpoint; (2) to propose assistive applications designed by human factors and HCI experts, in collaboration with caregivers and users; (3) to develop and test applications by software engineers; (4) to conduct a field study for assessing the benefits of the platform and assistive applications, in collaboration with caregivers, by deploying the system at the actual home of elders.

The HomeAssist platform is implemented on top of the DiaSuiteBox platform, using a suite of tools, namely DiaSuite, that have been designed, developed and tested by our research group at Inria. The DiaSuite tools include a dedicated integrated development environment that enables applications to be developed quickly and safely. This technology has been successfully applied to a variety of domains where environments consist of networked objects that need to be orchestrated.

5.4.1. Applications

HomeAssist offers an online catalog of applications. Using this catalog, the user and the caregiver determine what and how activities should be assisted by selecting the appropriate assistive applications and configuring them with respect to the user's requirements and preferences. The resulting set of applications forms a personalized assistive support. Additionally, to respond to evolving needs, our platform allows to stop/remove applications easily and to install new ones from the online catalog.

This platform proposes many applications in three domains of everyday life:

- Daily activities: including activity monitoring, light path, and a reminder.
- Home or personal safety: including entrance monitoring, stove monitoring, and warning if no movements are detected after a certain amount of time.
- Communications and social activities: including collaborative games, videoconference, information about local events, TV programming, *etc*.

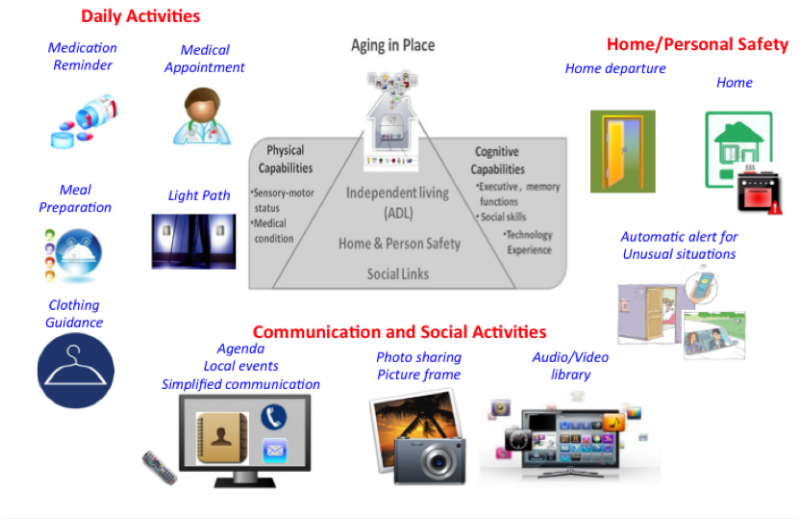

Figure 6. The HomeAssist platform and applications

For video presentations of HomeAssist, see the following:

- http://videotheque.inria.fr/videotheque/media/23705. Title: "Dia Suite Box". Produced in 2013.
- http://videotheque.inria.fr/videotheque/media/29998. Title: "DomAssist : L'assistance numérique à la personne". Produced in 2014.

5.4.2. Devices

Several entities have been identified to deliver an assistive support. These entities include (1) technological devices: wireless sensors (motion detectors, contact sensors and smart electric switches), and two tablets, and (2) software services (agenda, address book, mail agent, and photo agent) to monitor everyday activities and propose assistive applications. Sensors are placed in relevant rooms in the house: kitchen, bedroom, bathroom, and around the entrance.

Figure 7. HomeAssist devices

5.4.3. Experimental validation

A field study is currently being conducted with elderly people. The major purpose of this study is to identify the benefits of using HomeAssist for this population in an ecological framework. We selected 24 elderly people with different levels of autonomy (GIR scores). The HomeAssist technology has been installed in their house during 9 months. Twenty-four non-equipped elders were also selected to participate to the study, as control participants.

The expected impact of HomeAssist reflects the trans-disciplinary nature of the project. We aim to deliver results in the domain of (1) elderly care, (2) ergonomics and human factors, and (3) pervasive computing.

The major expected results are that HomeAssist (1) prolongs ageing in place, improves well-being of the users, and improves the efficiency of the caregiving environment; (2) is a cognitively low-cost assistive technology, and is well accepted and perceived as useful and usable by the users; (3) is technologically robust, and is a validated assistive platform

First results are expected in January 2015.

5.4.4. External Partners

The HomeAssist platform is being developed with support from the following partners:

- Équipe “Handicap et Système Nerveux” (EA 4136), Bordeaux University
- Chaire TSA, Université du Québec Trois-Rivières
- CRIUGM, Université de Montréal
- UDCCAS Gironde
- CARSAT
- Conseil Général 33
- Conseil Régional d’Aquitaine

5.5. DiaSwarm: a Development Environment for orchestrating smart objects at a large scale

Participants: Charles Consel [correspondent], Milan Kabac, Adrien Carteron, Eugène Volanschi.

The development of orchestrating applications which are responsible for large numbers of smart objects raises a number of challenges. We have addressed these by introducing a new design language called DiaSwarm, which is an extended version of the DiaSpec language.

5.5.1. Service discovery

Standard service discovery at the individual object level does not address the needs of applications orchestrating large numbers of smart objects. Instead, a high-level approach which provides constructs to specifying subsets of interest is needed. Our approach allows developers to introduce application-specific concepts (e.g., regrouping parking spaces into lots or districts) at the design time and then these can be used to express discovery operations. Following our design-driven development approach, these concepts are used to generate code to support and guide the programming phase.

5.5.2. Data gathering

Applications need to acquire data from a large number of objects through a variety of delivery models. For instance, air pollution sensors across a city may only push data to the relevant applications when pollution levels exceed tolerated levels. Tracking sensors, however, might determine the location of vehicles and send the acquired measurements to applications periodically (e.g., 10 min. intervals). Data delivery models need to be introduced at design time since they have a direct impact on the application's program structure. In doing so, the delivery models used by an application can be checked against sensor features early in the development process.

5.5.3. Data processing

Data that is generated from hundreds of thousands of objects and accumulated over a period of time calls for efficient processing strategies to ensure the required performance is attained. Our approach allows for an efficient implementation of the data processing stage by providing the developer with a framework based on the MapReduce [30] programming model which is intended for the processing of large data sets.

More details on this software platform can be found in the special issue on Smart Cities of the journal ERCIM News [18], 2014.

RMOD Project-Team

5. New Software and Platforms

5.1. Moose5.0

Participants: Stéphane Ducasse [correspondant], Muhammad Bhatti, Andre Calvante Hora, Nicolas Anquetil, Anne Etien, Guillaume Larcheveque, 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. The current focus is Moose 5.0, in late beta testing as of December 2014. A major development of 2014 is that tools and frameworks built for Moose are being integrated into Pharo4 as the default development tools.
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

5.2. Pharo3.0

Participants: Marcus Denker [correspondant], Damien Cassou, Stéphane Ducasse, Esteban Lorenzano, Damien Pollet, Igor Stasenko, Camillo Bruni, Camille Teruel, Clément Bera.

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

The platform. Pharo is an 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 applications, while at the same time continue to evolve.

The first stable version, Pharo 1.0, was released in 2010. We are now releasing one new version of Pharo every year, with Pharo3 released in spring 2014. Pharo4 has seen already over 400 incremental updates and is scheduled for early 2015. It should be noted that Pharo, even though already used outside of research, still continues to improve radically.

In November 2012 RMoD launched the Pharo Consortium (http://consortium.pharo.org/) and the Pharo Association (http://association.pharo.org). The consortium has now 14 industrial members, 3 sponsors and 10 academic partners.

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 in place.
- **(EM4)** Evolution and Maintenance: 4 – Active user group, consortium and association has 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

5.3. Pillar 0.17

Pillar is a markup syntax and associated tools to write and generate documentation and books. Pillar is currently used to write several books and other documentation. Two platforms have already been created on top of Pillar: PillarHub and Marina. http://www.smalltalkhub.com/#!/~Pier/Pillar

TACOMA Team

5. New Software and Platforms

5.1. THEGAME

Context-aware applications have to sense the environment in order to adapt themselves and provide with contextual services. This is the case of Smart Homes equipped with sensors and augmented appliances. However, sensors can be numerous, heterogeneous and unreliable. Thus the data fusion is complex and requires a solid theory to handle those problems. The aim of the data fusion, in our case, is to compute small pieces of context we call context attributes. Those context attributes are diverse and could be for example the presence in a room, the number of people in a room or even that someone may be sleeping in a room. For this purpose, we developed an implementation of the belief functions theory (BFT). THE GAME (THeory of Evidence in a lanGuage Adapted for Many Embedded systems) is made of a set of C-Libraries. It provides the basics of belief functions theory, computations are optimized for an embedded environment (binary representation of sets, conditional compilation and diverse algorithmic optimizations).

THE GAME is published under apache licence (https://github.com/bpietropaoli/THEGAME/). It is maintained and experimented by Aurélien Richez within a sensor network platform developed by TACOMA since June 2013.

COATI Project-Team

5. New Software and Platforms

5.1. JourneyPlanner

Participant: Marco Biazzini [correspondant].

JourneyPlanner is a Java implementation of a recursive algorithm to solve a TSP problem on small dense graphs, where non-trivial constraints must be satisfied, that make commonly used paradigms (as dynamic programming) unfit to the task.

This work is done in collaboration with the R&D service of the "Train Transportation" division of Amadeus.

5.2. Software updates

Participants: David Coudert, Luc Hogie, Aurélien Lancin, Nicolas Nisse, Michel Syska.

During this year, we have maintained and augmented already existing softwares. In particular:

- DRMSim (http://drmsim.gforge.inria.fr/) : discrete-event simulation engine aiming at enabling the large-scale simulations of routing models.
- GRPH (http://grph.inria.fr) : graph optimization library written in Java. This year, we have integrated the discrete-events simulation engine of DRMSim and some dynamic models (evolution of the connectivity with the mobility of nodes) to GRPH. Notice that we have identified more than 300 academic users of GRPH.
- Sage (http://www.sagemath.org) : open-source mathematics software initially created by William Stein (Professor of mathematics at Washington University). We contribute the addition of new graph algorithms along with their documentations and the improvement of underlying data structures.

5.3. Platforms

5.3.1. BigGraphs

Participants: Aurélien Lancin, Paul Bertot, Nicolas Chleq [SED-SOP], David Coudert, Luc Hogie, Fabrice Huet [Scale], Flavian Jacquot, Arnaud Legout [Diana], Eric Madelaine [Scale], Michel Syska [coordinator].

The objective of BigGraphs is to provide a distributed middleware for very large graphs processing. This new project has received a development grant (ADT) from Inria and is a joint work of three EPI from Inria: COATI, DIANA and SCALE.

The first phase of the project consists in the evaluation of the existing middlewares such as GraphX/Spark or Giraph/Hadoop with respect to the following criteria: ease of deployment, maintenance and use; variety of programming models (Map/Reduce, BSP, (a)synchronous message passing, centralized programming, mobile agent-based, etc.); overall efficiency and memory footprint; etc. One of the chosen use cases is a subgraph of the Twitter graph with 3 millions of nodes and 200 millions of edges. The experiments are run on the NEF cluster at Inria. We have implemented and tested the classic algorithms (using the BSP model): page rank, BFS, connected components as well as the iFUB algorithm for computing the diameter of large graphs.

In parallel, we are testing new ideas through the development of custom solutions for the deployment of application code in heterogeneous environments, for automatic discovery of cluster architecture, for the design of distributed object oriented applications, and techniques for distributed graph computing (asynchronous BSP, messaging, multi-core parallelism, etc.).

The next phase is to decide whether one framework is matching our needs (and use it as a basis for further developments) or if we have to produce our own.

DANTE Team

5. New Software and Platforms

5.1. Sensor Network Tools: drivers, OS and more

As outcomes of the Equipped FIT IoT-LAB, 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 IoT-LAB web site: https://www.iot-lab.info web page where one can find:

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

IoT-LAB software is licensed under a CeCILL License. IoT-LAB users are welcome to contribute code, papers, tutorials or experiments reports.

5.2. Queueing Systems

Online tool: http://queueing-systems.ens-lyon.fr

This tool aims at providing a simple web interface to promote the use of our proposed solutions to numerically solve classical queueing systems. It is a joint project between Thomas Begin (DANTE) and Pr. Brandwajn (UCSC). This tool supported since 2011 attracts each month hundreds of visitors from all around the world. Its current implementation includes the solution to:

- a queue with multiple servers, general arrivals, exponential services and a possibly finite buffer (*i.e.*, $Ph/M/c/N$-like queue) (refer to [32] for more details);
- a single server queue with Poisson arrivals, general services and a possibly finite buffer (*i.e.*, $M/Ph/1/N$-like queue);
- a queue with multiple servers, general service times and Poisson arrivals (*i.e.*, $M/Ph/c/N$-like queue) based on a recent work that was published in 2014 in Performance Evaluation [4]. Associated URL is: http://queueing-systems.ens-lyon.fr

DIANA Team

4. New Software and Platforms

4.1. ns-3

Participants: Walid Dabbous [correspondant], Thierry Turletti.

ns-3 is a discrete-event network simulator for Internet systems, targeted primarily for research and educational 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 realistic 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). ns-3 is free software, licensed under the GNU GPLv2 license, and it is publicly available for research, development, and use. Our team has been involved in ns-3 project since 2006 and we are founding member of the ns-3 consortium.

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

- Version: ns-3.21
- 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

4.2. DCE

Participants: Thierry Turletti [correspondant], Walid Dabbous.

DCE enables developers and researchers to develop their protocols and applications in a fully controllable and deterministic environment, where tests can be repeated with reproducible results. It allows unmodified protocol implementations and application code to be tested over large and possibly complex network topologies through the ns-3 discrete-event network simulator. The single-process model used in the DCE virtualization core brings key features, such as the possibility to easily debug a distributed system over multiple simulated nodes without the need of a distributed and complex debugger. Examples of tested applications over DCE include Quagga, iperf, torrent, thttpd, CCNx and various Linux kernel versions (from 2.6.36 to 3.12 versions). DCE was initially developed by Mathieu Lacage during his PhD thesis and is maintained by engineers in the team in collaboration with Hajime Tazaki from University of Tokyo. DCE/ns-3 is an important component of the Reproducible Research Lab. DCE is free software, licensed under the GNU GPLv2 license, and is publicly available for research, development, and use.

See also the web page https://www.nsnam.org/overview/projects/direct-code-execution/

- Version: DCE-1.2
- Keywords: emulation, virtualization, networking event-driven simulation
- License: GPL (GPLv2)
- Type of human computer interaction: programmation C/C++, No GUI
- OS/Middleware: Linux
- Required library or software: standard C++ library: GPLv2
- Programming language: C++, python
- Documentation: doxygen

4.3. NEPI

Participants: Thierry Turletti [correspondant], Alina Quereilhac, Julien Tribino, Lucia Guevgeozian Odizzio.

NEPI, the Network Experimentation Programming Interface, is a framework to describe and orchestrate network experiments on a variety of network experimentation platforms, including simulators, emulators, live testbeds, and testbed federations. NEPI is capable of supporting arbitrary platforms through the use of a generic network experiment description model, based on abstracting network experiments as a collection of arbitrary resource objects, and through the generalization of the experiment life cycle for all resources. The common resource life cycle consist on the sequence of operations deploy, start, stop, and release. Different resource objects can implement specific versions of those operations to adapt to any platform. NEPI resolves experiment orchestration as an online scheduling problem that consists on executing the deploy, start, stop, and release operations for every resource in the correct order.

During the year 2013 we fully re-implemented NEPI's core libraries to adopt the scheduling-based experiment orchestration approach, improving the flexibility and extensibility of the framework compared to the previous static stage-based orchestration approach. By the end of 2013 the new NEPI framework supported describing and orchestrating experiments on live testbeds, including SSH enables Linux testbed, PlanetLab Internet testbed, and OMF wireless testbed (version 5.4).

In 2014 the framework was extended to support simulation and emulation, using the ns-3 simulator and its direct code execution (DCE) emulation extension. Additionally, automated translation of a same experiment scenario to different platforms, i.e. multi platform experimentation, was incorporated into the framework to provide an unified environment for the development and evaluation of production quality networking software, meant to be deployed on real networks. This unified development and evaluation environment simplifies the transition from a realistic live platform to a controlled emulation platform, and vice-versa, in order to take advantage of the complementary features offered by them. The environment was demonstrated at ACM ICN 2014 for the case of Content Centric Networking (CCNx) development, combining PlanetLab and DCE platforms.

Finally, NEPI now supports OMF experiment control protocol version 6.0, which is the new mainstream release of OMF control framework for testbeds, and SFA (Slice Federation Architecture), for resource discovery and provisioning across federated testbeds. The combination of OMF 6.0 and SFA was adopted as the standard for federated experiment orchestration in the European federation projects OpenLab and Fed4FIRE. NEPI's ability to support federated experiment orchetsration was demonstrated at the OpenLab Final Review. Information about this demo is available at http://nepi.inria.fr/UseCases/VLCCCNStreamingExperiment.

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

- Version: 3.2
- ACM: C.2.2, C.2.4
- Keywords: networking experimentation, simulation, emulation
- License: GPL (3)
- Type of human computer interaction: python library
- OS/Middelware: Linux
- Required library or software: python – http://www.python.org
- Programming language: python

4.4. OpenLISP

Participant: Damien Saucez [correspondant].

Among many options tackling the scalability issues of the current Internet routing architecture, the Locator/Identifier Separation Protocol (LISP) appears as a viable solution. LISP improves a network's scalability, flexibility, and traffic engineering, enabling mobility with limited overhead. As for any new technology, implementation and deployment are essential to gather and master the real benefits that it provides. We propose a complete open source implementation of the LISP control plane. Our implementation is deployed in the worldwide LISP Beta Network and the French LISP-Lab testbed, and includes the key standardized control plane features. Our control plane software is the companion of the existing OpenLISP dataplane implementation, allowing the deployment of a fully functional open source LISP network compatible with any implementation respecting the standards.

See also the web page http://www.lisp.ipv6.lip6.fr/a/Download.html.

- Version: 3.2
- ACM: C.2.1, C.2.2, C.2.6
- Keywords: routing, LISP, control-plane
- License: BSD
- Type of human computer interaction: XML, CLI
- OS/Middelware: POSIX
- Required library or software: Expat 2
- Programming language: C
- Documentation: Unix man
- Deployment: ddt-root.org

4.5. ACQUA

Participants: Chadi Barakat [correspondant], Salim Afra, Damien Saucez.

ACQUA is an Application for Predicting User Quality of Experience at Internet Access. It was supported by the French ANR CMON project on collaborative monitoring. ACQUA presents a new way for the evaluation of the performance of Internet access. Starting from network-level measurements as the ones we often do today (bandwidth, delay, loss rates, etc), ACQUA targets the estimated quality of experience related to the different applications if run at the access. An application in ACQUA is a function that links the network-level measurements to its expected quality of experience. In its first version (the version available online), ACQUA was concentrating on delay measurements at the access and on the detection and estimation of the impact of delay anomalies (local problems, remote problems, etc). The current work is concentrating on using the ACQUA principle in the estimation and prediction of the quality of experience of main applications (see section 5.2 for more details).

See also the web page https://team.inria.fr/diana/acqua/.

- Version: 1.0
- ACM: C.2.2, C.2.3
- Keywords: Internet measurement, Internet Access, Quality of Experience
- License: GPL (3)
- Type of human computer interaction: C#
- OS/Middelware: MS Windows
- Required library or software: visual studio http://www.visualstudio.com/en-us/products/visual-studio-express-vs.aspx
- Programming language: C# for client, java for server

4.6. ElectroSmart

Participants: Arnaud Legout [correspondant], Inderjeet Singh, Maksym Gabielkov.

The ElectroSmart project is based on a large crowd sourcing collection of electromagnetic radiations measured by the ElectroSmart application running on real users smartphones. We target a large number of users and many scientific exploitation of the collected data, exploitation that we describe in the following.

Exposure of human beings to electromagnetic radiations is a growing worldwide health concern. While the biological impact of electromagnetic radiations is not fully understood, there are reports of hypersensitivity to such radiations and hints toward a possible correlation between high exposition and cancer. However, the biological impact of electromagnetic radiations is just one half of the problem, the other half is the exploration of the real exposure of the population to electromagnetic radiations. Indeed, whatever the biological impact, it will be function of the level of exposure, and this level of exposure is unknown.

Collecting the real exposure of human beings to electromagnetic radiations is a complex task. It is possible, but costly and time consuming, to ask auditing organizations to make one-shot measurements. However, there is no way accessible to the general audience to make long term measurements.

The goal of this project is to create the first long term measurement of the electromagnetic exposure of a large worldwide population. This project is supported by the Inria ADT ElectroSmart.

- Version: 1.0alpha
- Keywords: background electromagnetic radiations
- License: Inria proprietary licence
- Type of human computer interaction: Android application
- OS/Middleware: Android
- Required library or software: Android
- Programming language: Java
- Documentation: javadoc

4.7. Platforms

4.7.1. Reproducible research laboratory ($\mathrm{R^2LAB}$*)*

Scientific evaluation of network protocols requires that experiment results must be reproducible before they can be considered as valid. This is particularly difficult to obtain in the wireless networking domain, where characteristics of wireless channels are known to be variable, unpredictable and hardly controllable. Indeed, anechoic chambers with RF absorbers preventing radio waves reflections and with Faraday cage blocking external interferences represent an ideal environment for experiments reproducibility. This year witnessed the realization of such experimental platform (called $\mathrm{R^2LAB}$ or Reproducible Research Laboratory) at Inria Sophia-Antipolis, in the context of the FIT 'Equipment of Excellence' project. The objectives of this platform are twofold : on the one hand, we need to achieve highly controllable wireless experiments (e.g. control plane for 5G), and to this end, the testbed features an anechoic chamber. On the other hand, we need to make it possible to deploy experiments that have demanding resource requirements, as this is typically the case with e.g. ICN-based research, or when involving simulation. For that reason, the platform features some powerful servers of its own; in addition, these experiments can be either hybrid-experiments (as NEPI will be deployed) or federated experiments through several testbeds such as PlanetLab. As the final objective is to provide an environment to easily run realistic and reproducible wireless experiments and simulations, it is important to be able to increase the testbed realism by injecting noise and interfering signals in a controllable way. Experimentation results done in $\mathrm{R^2LAB}$ could also be used to augment the realism of propagation models in simulators, which are able to run large scale scenarios. We are currently deploying the wireless nodes. The next step will to extend the testbed to support software defined networking (SDN) and LTE experimentations and to install specific tools for operating the platform.

DIONYSOS Project-Team

4. New Software and Platforms

4.1. T3devKit testing toolkit and IPv6 test suites

Participant: 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

Participant: 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 and (ii) facilitating new approaches and methods for interoperability assessment. For instance, new passive approaches 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. 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, etc. A set of functions designed for dependability analysis is being built under the name `DependLib`.

Pierre L'Ecuyer is also developing in Montreal a library, *Stochastic Simulation in Java* (SSJ), providing facilities for generating uniform and nonuniform random variates, computing different measures related to probability distributions, performing goodness-of-fit tests, applying quasi-Monte Carlo methods, collecting (elementary) statistics, and programming discrete-event simulations with both events and processes.

DYOGENE Project-Team

5. New Software and Platforms

5.1. Clones: CLOsed queueing Networks Exact Sampling

Clones is a Matlab toolbox for exact sampling of closed queueing networks.

Details can be found in [18] (best tool-paper award).

Available at: http://www.di.ens.fr/~rovetta/Clones/index.html

FUN Project-Team

4. New Software and Platforms

4.1. New ALE module for ASPIRERFID middleware.

Participants: Rim Driss [correspondant], Nathalie Mitton, Ibrahim Amadou, Julien Vandaele.

AspireRFID middleware is a modular OW2 open source RFID middleware. It is compliant with EPC Global standards. This new module integrates the modifications of the new standard release, including new RP and LLRP definitions and fixing bugs. This module has been implemented in the framework of the MIAOU project.

- Version: 1.0. . APP number: IDDN.FR.001.100017.000.S.P.2012.000.10000

4.2. T-SCAN.

Participants: Gabriele Sabatino [correspondant], Nathalie Mitton.

T-Scan is an interface ensuring the translation from a SGTIN tag format to an ONS hostname format according to the EPCGlobal standards. It allows the sending of a DNS request to look up the EPC-IS aides to which the product belongs in order to access the data relative to that product. This module has been implemented in the framework of the TRACAVERRE project.

- Version: 1.0 . March 1st 2014. N IDDN abrégé : 14-440017-000

4.3. GOLIATH 1.0

Participants: Nathalie Mitton [correspondant], Salvatore Guzzo Bonifacio [correspondant].

GOLIATH (Generic Optimized LIghtweight communication stack for Ambient TecHnologies) is a full protocol stack for wireless sensor networks. This module has been implemented in the framework of the ETIPOPS project.

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

4.4. ETINODE-CONTIKI-PORT

Participants: Salvatore Guzzo Bonifacio [correspondant], Roudy Dagher, Nathalie Mitton.

Contiki is an open source embedded OS for Internet of Things (IoT). It is light and portable to different hardware architectures. It embeds communication stacks for IoT Il embarque aussi des piles de communication pour l'internet des objets. This driver allows the running of Contiki OS over Etinode-MSP430. The code dalso allows the use of radio chip and embedded sensors. This module has been implemented in the framework of the ETIPOPS project.

- Version: 1.0 .

4.5. ETINODE-DRIVERS

Participants: Salvatore Guzzo Bonifacio [correspondant], Roudy Dagher, Nathalie Mitton.

These drivers for Etinode-MSP430 control the different embedded sensors and hardware components available on an Etinode-MSP430 node such as gyroscope, accelerometer and barometric sensor. This module has been implemented in the framework of the ETIPOPS project.

- Version: 1.0 .

4.6. FIT IoT-Lab

Participants: Raymond Borenstein, Nathalie Mitton [correspondant], Anne-Sophie Tonneau, Julien Vandaele, Roberto Quilez.

FIT IoT-LAB is a very large scale open testbed that features over 2700 wireless sensor nodes and more than 200 robots spread across six different sites in France. Nodes are either fixed or mobile and can be allocated in various topologies throughout all sites. A variety of wireless sensors are available, with different processor architectures (MSP430, STM32 and Cortex-A8) and different wireless chips (802.15.4 PHY at 800 MHz or 2.4 GHz). In addition, "open nodes" can receive custom wireless sensors for inclusion in IoT-LAB testbed. This platform is completely open and can be used by any one wishing to run experiment on wireless sensors and robots.

The Lille site displays 3 subsets of the platforms:

- Euratechnologies : this site features 256 WSN430 sensor nodes operating in the 2.4GHz band. 64 nodes are mobile, embedded on mobile trains.
- Haute Borne : this site features 256 M3 sensor nodes operating in the 2.4GHz band and 64 mobile robots (32 turtlebots and 32 wifibots) completely remotely programmable.
- Opennodes : this site will feature (opening beginning 2015) 64 hardware open slots to allow any one to plug his own hardware and benefits from the platform debugging and monitoring tools.

GANG Project-Team (section vide)

HIPERCOM2 Team

5. New Software and Platforms

5.1. OPERA and OCARI Software

Participants: Cédric Adjih, Ichrak Amdouni, Pascale Minet, Ridha Soua, Erwan Livolant.

The OPERA software was developed by the Hipercom2 team in the OCARI project (see https://ocari.org/). It includes EOLSR, an energy efficient routing protocol and OSERENA, a coloring algorithm optimized for dense wireless networks. It was registered by the APP. In 2013, OPERA has been made available for download as an open software from the InriaGForge site: https://gforge.inria.fr/scm/?group_id=4665

In 2014, OPERA has been ported on a more powerful platform based on the Atmel transceiver AT86RF233 and on a 32 bits microcontroler Cortex M3.

More details and documentation about this software are available in the website made by the Hipercom2 team: http://opera.gforge.inria.fr/index.html

Erwan Livolant, Pascale Minet from Inria as well as Tuan Dang from EDF and Maurice Sellin from DCNS showed the wireless sensor network OCARI during the Inria-Industry Meeting devoted to Telecommunications organized by Inria at Rocquencourt in November 2014. Two types of demonstration were done: one illustrating the internal behavior of OCARI and the other one illustrating a fire detection in a DCNS ship.

5.2. SAHARA Software

Participants: Erwan Livolant, Pascale Minet, Ridha Soua.

The software module DimTool developed in 2014 by the Hipercom2 team in the SAHARA project has been registered by the APP in January 2014. It helps to dimension the network parameters of the SAHARA wireless sensor network and evaluate the feasibility of a given application.

5.3. CONNEXION Software

Participants: Ines Khoufi, Pascale Minet, Erwan Livolant.

In 2014, we developped two softwares to compute the positions of:

- sensor nodes that ensure full coverage of a 2-D area with irregular shape and containing obstacles.
- relay nodes that maintain a robust connectivity of each point of interest with the sink. The area may contain obstacles.

With regard to the wireless sensor network OCARI, in 2014 we developed the interface of wireless temperature sensors PT100 with the OCARI stack. We also gave our support to CEA for the development of the OCARI interface of smoke detectors. With Telecom ParisTech, we interconnected OCARI with the industrial facility backbone based on OPC/UA via a gateway implemented on a Raspberry Pi. More precisely, we developed the serial interface between the OCARI network coordinator and the OPC/UA server.

5.4. NS3 Network Coding Software

Participants: Cédric Adjih, Ichrak Amdouni, Hana Baccouch.

One output of the GETRF project, was the creation of a solution for Network Coding, called DragonNet. DragonNet is a complete modular solution. This solution is responsible of: coding, decoding, maintaining necessary information and the associated signaling. It is designed to be extensible. A variant of DragonNet has been specified for wireless sensor networks and implemented.

As a follow-up to the ADT MOBSIM (and the previous module EyWifi), DragonNet was also integrated as a module for the ns-3 simulation tool.

5.5. FIT IoT-LAB Platforms

Participants: Cédric Adjih, Alaeddine Weslati, Vincent Ladeveze, Ichrak Amdouni.

This is a joint work with Emmanuel Baccelli from Inria Saclay.

Period: 2011 - 2021

Partners: Inria (Lille, Sophia-Antipolis, Grenoble), INSA, UPMC, Institut Télécom Paris, Institut Télécom Evry, LSIIT Strasbourg.

- Deployment: During the year 2014, the practical deployment has been finished, at the location planned for this testbed of the EQUIPEX FIT: the basement of building 1 at Rocquencourt. Ten external nodes have also been integrated.

  The testbed is offering most of the planned 344 open nodes, including 120 WSN430 nodes, 200 Cortex A8 based nodes, 24 Cortex M3.
- Opening: The official opening of the Rocquencourt was done in November 2014, at that point all IoT-LAB (and OneLab) users could run experiments in the M3 and A8 nodes from the site. See https://www.iot-lab.info/deployment/rocquencourt/ for more information.
- Support of external projects: Support for RIOT-OS and OpenWSN projects has been developed for IoT-Lab hardware and is being tested.
  - RIOT-OS, a joint work between Inria and FU-Berlin to create an Operating System for the Internet of Things.
  - OpenWSN, an open-source protocol stack for Internet of Things developed by UC Berkley.
- Demonstration: the project IoT-LAB in general had been demonstrated in several events during the year
  - March 2014: Description of IoT-LAB, and test of IoT-LAB nodes at the IETF 6TiSCH plugfest during IETF-89 in London https://bitbucket.org/6tisch/meetings/wiki/140306a_ietf89_london_plugfest
  - July 2014: Demonstration of IoT-LAB (Contiki RPL experiments) at:
    * the Bits-n-Bytes event of IETF-90 in Toronto http://www.ietf.org/meeting/90/ietf-90-bits-n-bites.html
    * the LLN plugfest of IETF-90 in Toronto https://bitbucket.org/6tisch/meetings/wiki/140720a_ietf90_toronto_plugfest
  - November 2014: Demonstration of IoT-LAB for use of scientific experiments, with network coding at MASS 2014 (http://mass2014.eecs.utk.edu/)

Moreover, during 2014, Ichrak Amdouni was in charge of:

- Testing the support of new switches in FIT IoT-LAB Paris-Rocquencourt site.
- Experimenting network coding protocols in the FIT IoT-LAB platform.

INFINE Team

5. New Software and Platforms

5.1. Software

5.1.1. MACACOapp

Participant: Aline Carneiro Viana.

MACACOapp (https://macaco.inria.fr/macacoapp/) is developped in the context of the EU CHIST-ERA MACACO project (https://macaco.inria.fr/). It consists in a mobile phone application that periodically samples phone's information on the mobility (through, e.g., GPS sensor, accelerometer and WiFi/Bluetooth/Cellular environment, connectivity type) and on the data traffic it generates (through, e.g., Internet browser history and applications data consumption). The information collected will be time-stamped and will be periodically sent to the central servers for analysis and visualization. We expect that (1) the collected information will allow us studying the correlation between mobility and content demand patterns and that (2) the results of this analysis will allow us inferring the best times and places to transfer content from/to users' phones location and/or from/to the wireless infrastructure closest to the users' phones location. Users will be also invited to fill a non-mandatory questionnaire relevant to this study. Our questionnaire collects information about the personality traits and application preferences of people. We expect that the information collected from questionnaire will allow us to analyse the correlation between users personality traits and their application preferences and interests. User's application preferences and interests will be inferred from the Internet browsing history and running app information obtained from the MACACO App.

The data collection and on-the-phone storage of MACACOapp is designed in accordance with the state-of-the-art best practices in application development. The data collected on the phone is encrypted and inaccessible by any other application installed on the same phone or to any other third party, even in case your phone gets lost or stolen. Moreover, any user's identity information available in the collected data or in the questionnaire will be completely and irreversibly anonymised before its transfer to the central servers. The on-the-phone collected data and questionnaire data will be transferred via a secure transmission protocol to the central servers. This application is in phase of test.

5.1.2. RIOT

Participants: Emmanuel Baccelli, Oliver Hahm.

RIOT (http://www.riot-os.org) is a nano operating system for the Internet of Things. While requiring as low as 1,5kB of RAM and 5kB or ROM, RIOT offers real time and energy efficiency capabilities, as well as a single API (partially POSIX compliant) across heterogeneous 8-bit, 16-bit and 32-bit low-hardware. This API is developer-friendly in that it enables multi-threading, standard C and C++ application programming and the use of standard debugging tools (which was not possible so far for embedded programming). On top of this, RIOT includes several network stacks, such as a standard IPv6/6LoWPAN stack and a information-centric network stack (based on CCN).

RIOT is developed by an international community of open-source developers that was co-founded by Inria and Freie Universitaet Berlin. The goal of RIOT is to provide a powerful, free, open-source IoT software platform that can be used like Linux is for less constrained machines, both (i) in the context of research and/or teaching, as well as (ii) in industrial contexts.

5.1.3. DragonNet

Participants: Cédric Adjih, Ichrak Amdouni, Hana Baccouch, Antonia Masucci.

DragonNet is a generic framework for network coding in wireless networks. It is a initially result of the GETRF project of the Hipercom2 team.

It is based on intra-flow coding where the source divides the flow in a sequence of payloads of equal size (padding may be used). The design keys of DragonNet are simplicity and universality; DragonNet does not use explicit or implicit knowledge about the topology (such as the direction or distance to the source, the loss rate of the links, ...). Hence, it is perfectly suited to the most dynamic wireless networks. The protocol is distributed and requires minimal coordination. DragonNet architecture is modular, it is based on 5 building blocks (LIB, SIG, Protocol, SEW and DRAGON). Each block is almost independent. This makes DragonNet generic and hence adaptable to many application scenarios. DragonNet derives from a prior protocol called DRAGONCAST. Indeed, DragonNet shares the same principles and theoretical overview of DRAGONCAST. It enriches DRAGONCAST by the information base and signaling required to perform broadcast in wireless networks and in wireless sensor networks in particular.

5.2. Platforms

5.2.1. FIT IoT-LAB

Participants: Cedric Adjih, Emmanuel Baccelli, Ichrak Amdouni.

FIT IoT-LAB is a platform built to help foster the development, tuning and experimentation of protocols and applications for the Internet of Things and wireless sensor networks. IoT-LAB provides both dedicated IoT hardware deployments, a front-end webportal and backend management software. Using these elements, IoT-LAB enables users to share access to this IoT hardware, set-up and manage experiments. Remote use, and large scale experiments on concrete IoT deployments are thus made possible.

The Infine team is now managing the IoT-LAB site currently located in Rocquencourt, and which was publically opened in November 2014. It consists of the following:

- A set of GPS repeaters are relaying the GPS signal indoor (used for time synchronization)
- 200 A8 nodes, all equipped with GPS (10 deployed outside – identifiers between 166 and 175)
- 24 M3 nodes
- 120 WSN430 nodes

This platform was developped as part of the Equipex FIT (see section 8.1.1).

MADYNES Project-Team

5. New Software and Platforms

5.1. Escape

Participants: Thibault Cholez [contact], Shbair Wazen.

Initially developed by Antoine Goichot during his internship [47], from reasearch results of Wazen Shbair, Thibault Cholez and Isabelle Chrisment.

Escape is a Firefox web browser addon designed and developed by the team to bypass some HTTPS filtering strategies. The extension was built in the context of evaluating HTTPS traffic filtering techniques based on the Server Name Indication (SNI) extension of TLS and which have been recently used by many firewalls for filtering websites accessed through HTTPS. Our tool mainly offers the ability to bypass such firewalls by editing on-the-fly the SNI field with alternate values and therefore access the blocked HTTPS websites. In addition, it can be used to bypass legacy filtering of DNS requests. The extension is implemented in JavaScript and is based on another security addon named Convergence. Escape is distributed under a GPL3 Open Source license and can be downloaded on the team website.

5.2. MPIGate

Participants: Mandar Harshe, Ye-Qiong Song [contact].

MPIGate stands for Multi Protocol Interface GATEway for Tele-care, Environment Monitoring and Control. It was initiated by TRIO Team of LORIA and Inria Nancy Grand-est, in October 2009 as a follow-up of wireless sensor network (WSN) projects in ambient assisted living, smart home, logistic and industry domains. Since 2012, its evolution is continuously ensured by members of MADYNES Team. It is a set of software aiming at facilitating the development of both home automation and ambient assisted living applications thanks to the abstraction of heterogeneous sensor data and the facility of access to read and write functions over the devices plugged to the networks (wired and wirelessly). The key features of MPIGate include the drivers for different networks protocols (Bluetooth, WiFi, IEEE802.15.4/Zigbee, KNX, EnOcean) and a ROS-based middleware layer offering modularity and quality of service. This year, its evolution has mainly been carried out within SATELOR project and IPL PAL project. It can be used by people working on home automation and ambient assisted living applications. Further information can be found at http://mpigate.loria.fr.

5.3. AA4MM

Participants: Laurent Ciarletta [contact], Yannick Presse, Benjamin Segault.

Benjamin Camus, Victorien Elvinger, Vincent Chevrier (contact), Julien Vaubourg, and Christine Bourjot from the MAIA team, LORIA are contributors for this software.

AA4MM (Agents and Artefacts for Multi-modeling and Multi-simulation) is a framework for coupling existing and heterogeneous models and simulators in order to model and simulate complex systems. The first implementation of the AA4MM meta-model was proposed in Julien Siebert's PhD [51] and written in Java, and a renewed Java version was submitted to the APP (Agence pour la protection des programmes).

We are using this software in a strategic action with EDF R&D in the context of the simulation of smart-grids in the frame of the MS4SG (Multi-Simulation fro Smart Grids) project. Julien Vaubourg started a PhD on this project that is co-directed by Laurent Ciarletta and Vincent Chevrier. The 2014 year was dedicated to improve existing software and to develop new components thanks to new scientific contributions.

Currently, two new pieces of software are being submitted to the APP:

1. a modelling environment software that enables the graphical definition of multi-models from preexisting elements.
2. AA4MM-Visu, a plug-in dedicated to the collection and visualization of information during simulation.

We plan to submit an enhanced version of the JAVA software and of the AA4MM-Visu. The core elements of AA4MM will be made available early in 2015 under an open licence.

5.4. Platforms

5.4.1. Android Security platform

Participants: Abdelkader Lahmadi [contact], Rémi Badonnel, Olivier Festor, Eric Finickel, Frederic Beck [SED, Inria Nancy Grand Est].

Android environments are facing several threats and attacks. Madynes team is working on the development of a monitoring platform dedicated to the security analysis and these environments. The monitoring platform relies on different components:

- a set of probes dedicated to the measurement of network activities using NetFlow protocol and logs generated by running Applications of an Android device. An OVAL agent (Ovaldroid) is also developed for vulnerability assessment.
- a set of scalable data collectors to collect and parse the data issued by our probes (NetFlow records, logs in the syslog format and vulnerability reports). The collectors are relying on Flume agents.
- a NoSQL storage (HBase) engine where all the collected data are stored for further analysis.
- A first set of analysers of the collected data, relying on a Map-Reduce engine (Spark) are also developed [41] including statistical analysis about connected services and ports but also a Self-Organising Map analyser to classify Android application patterns according to different properties including their communication patterns and also their lifecycle activities. [16].

The first version of the monitoring platform is operational and deployed within the LHS infrastructure. Further development is currently under taken to provide more analysis, data correlation and visualisation features.

5.4.2. IoT platform

Participants: Emmanuel Nataf [contact], Thibault Cholez.

This platform is a joint work between Anthony Deroche [43], Thierry Duhal [45] and Arthur Garnier [46], respectively students from TELECOM Nancy and IUT Nancy-Charlemagne. They worked under the supervision of Emmanuel Nataf and Thibault Cholez between February and August 2014.

The main goal of the IOT platform is to collect and store production and management data produced during long-run WSN experiments. The platform is open-source (https://github.com/AnthonyDeroche/iotlab/) and built with a modular architecture in order to support different types of experiment (routing algorithms, energy efficiency, security, etc.).

Based on this platform, we developed several innovative applications:

- indoor geolocalization of sensors based on RSSI strength [43]
- data collection from several concurrent points allowing better scalability with good performances on large WSN [45]
- data link to remotely control nodes from the web interface with a skeleton of API [45]

Regarding the technical aspects [44], the platform is based on a JEE architecture running on a Glassfish server, websocket full-duplex communications, secure and authenticated administrator access (HTTPS). The web interface uses the framework CSS front-end Zurb Foundation and javascript libraries to display dynamic charts and maps.

The full plateform has been instantiated with 40 TELOSB sensors deployed in TELECOM Nancy (http://iotlab.telecomnancy.eu/) during one month.

5.4.3. SCADA platform

Participants: Abdelkader Lahmadi [contact], Jérôme François, Olivier Festor.

SCADA is a term used in several industries and its stands for *Supervisory Control and Data Acquisitions*. It refers to a centralized control and monitoring system for a variety of machinery and equipment involved with many industrial activities. SCADA systems are also becoming target to different attacks exploiting traditional IT vulnerabilities, e.g. buffer overflows, script crossing, crafted network packets, or specific vulnerabilities related to control and estimation algorithms employed by control processes.

We are developing and maintaining a platform to assess and analyse the security of SCADA systems based on a testbed combining real hardware and simulation tools of physical processes. We have extended our SCADA testbed to simulate a microgrid scenario [49]. We are thus able to extract and analyse the Profinet messages at the control network level using process mining techniques. Further development will be taken to include information technology layers in the testbed (servers, firewalls, network devices, etc).

During the year 2014, we have also started the development of a scanning platform of Internet IP addresses and communication ports to identify exposed sensitive services and networks, for instance SCADA systems [42].

MAESTRO Project-Team

5. New Software and Platforms

5.1. New Software

5.1.1. ns-3

Participants: Sara Alouf, Abdulhalim Dandoush, Giovanni Neglia.

ns-3 is an open source, C++ based, GPL licensed and highly used discrete-event network simulator. It is targeted primarily for research and educational use. ns-3 is particularly suited for the goals of the research project with ALSTOM Transport (see §7.1.3). New modules have been developed to enable the simulation of the real antennas used by ALSTOM. Also, modules related to the handoff procedure were debugged and modified to fit the proprietary algorithm used by ALSTOM. Another new module allows to simulate the proprietary communication-based train control protocol used by ALSTOM.

5.2. Platforms

5.2.1. Marmote

Participants: Alain Jean-Marie, Issam Rabhi.

In the framework of the ANR MARMOTE, a new software platform dedicated to Markovian modeling is being built. The architecture has been defined so as to be compatible with the software previously developed by members of the project, principally PSI (from the team MESCAL, joint between Inria, Univ. Joseph Fourier (Grenoble) and Institut polytechnique de Grenoble) and XBORNE (from the MAGMAT team of the Univ. Versailles St Quentin). The platform will provide a user interface allowing the modeler to access to a large number of solution methods for generic Markov models as well as optimized methods for specific families of models.

MUSE Team

5. New Software and Platforms

5.1. Fathom v2.0

Contributors: Anna-Kaisa Pietilainen, Stephane Archer

Available at: https://muse.inria.fr/fathom

Fathom [9] is a Firefox browser extension that explores the browser as a platform for network measurement and troubleshooting. It provides a wide range of networking primitives directly to in-page JavaScript including raw TCP/UDP sockets, higher-level protocol APIs such as DNS, HTTP, and UPnP, and ready-made functionality such as pings and traceroutes. Fathom v2.0 is a complete rewrite of the original Fathom using the new add-on SDK from Mozilla. In addition to javascript APIs, we have improved and added new built-in network measurement tools to Fathom such as 'Debug my Connection', 'Homenet Discovery' and 'Baseline Monitoring' that allow users to troubleshoot home network problems and share with us data for further research on home networks usage and diagnosis.

5.2. OpenWRT Packages for Network Measurements

Contributors: Anna-Kaisa Pietilainen, Sarthak Grover

Available at: https://github.com/apietila/browserlab

OpenWRT is a version of the Linux operating system to run on embedded devices, in particular, in home routers and access points. We have developed an OpenWRT package repository that provides fixes for and new ports of several existing network measurement tools (including iperf, iperf3, shaperprobe and pathload) for OpenWRT and an extended JSON RPC API for LuCI (OpenWRT control interface) to collaborate with the Fathom extension on home network diagnosis.

5.3. TagIt

Contributors: Sara El Aouad, Christophe Diot (Technicolor), Renata Teixeira

Available at: https://drive.google.com/file/d/0B-OcOkKOXok2b3lZYmt4QUw0cGM/view?usp=sharing
Video demo available at:
https://drive.google.com/file/d/0B-OcOkKOXok2VUxQR1NmRlg5VE0/view?usp=sharing

TagIt is an android app that makes it easy for users to enter movie reviews and to summarise the set of reviews of each movie. TagIt uses tags (or a short sequence of words) for entering user's opinions about a movie, so it is easy and quick for users to enter their feedback. TagIt also allows users to quickly see the opinion of other users about a movie with a tag cloud that summarises the set of reviews of a movie.

5.4. Where is the fault?

Contributors: Srikanth Sundaresan (Georgia Tech), Nick Feamster (Georgia Tech), Renata Teixeira

Where's The Fault? (WTF) is a system that localizes performance problems in home and access networks. We implement WTF as custom firmware that runs in an off-the-shelf home router. WTF uses timing and buffering information from passively monitored traffic at home routers to detect both access link and wireless network bottlenecks. We presented a demo of WTF at the ACM SIGCOMM conference in 2014. [4]

5.5. WeBrowse

Contributors: Giuseppe Scavo, Zied Ben Houidi (Alcatel-Lucent Bell Labs), Stefano Traverso (Politecnico di Torino), Marco Mellia (Politecnico di Torino), Renata Teixeira

Available at: http://tstat.polito.it/netcurator/

WeBrowse is the first passive crowdsource-based content curation system. Content curation is the act of assisting users to identify relevant and interesting content in the Internet. WeBrowse requires no active user engagement to promote content. Instead, it extracts the URLs users visit from traffic traversing an ISP network to identify popular content. WeBrowse contains a set of heuristics to identify the set of URLs users visit and to select the subset that are interesting to users [7]. The system proposes the interesting content in a web page available to all users.

5.6. UCN Data Collection

Contributors: Anna-Kaisa Pietilainen, Tom Logde (University of Nottingham), Richard Mortier (University of Nottingham), Peter Tolmie (University of Notthingham), Renata Teixeira

Available at: https://muse.inria.fr/ucn, code https://github.com/ucn-eu

The User-Centric Networking (UCN) project is seeking to understand how people consume various kinds of content when using computer networks. Within this project we are undertaking a detailed user study across a range of environments in order to understand the practices involved in consuming media and other content according to context. For the study, we have set up the following tools and software:

- **Registration and management website:** we have developed a website containing information about the experiment, and user and device registration interfaces.
- **VPN server and clients for network traffic data collection:** we are using OpenVPN open-source VPN server and available free clients on multiple platforms (OpenVPN for Linux, OpenVPN for Android, Tunnelblick for OS X, OpenVPN Connect for iOS) to collect network traffic traces from the participating devices. The VPN server is running on a secure Inria server, and we collect packet headers using tcpdump and http traffic logs with a Squid HTTP proxy. Collected data is stored on another server not directly accessible from the Internet.
- **Activity logging software:** we have developed a small Android application to log additional activity details such as list of running applications, foreground application, screen state, network connectivity details, and system resources (cpu, memory, network, battery) usage.
- **Data collection from Moves and Google Calendar:** we have written some code to import user data from Moves application and a Google Calendar based diary to add user location and daily activity logs to the data set.
- **Data visualisation:** the website contains a section to visualise all the collected data (network traffic as a function of location, time of day, activity) to support interviews with an ethnographer.

  We have obtained Ethics approval from Inria's COERLE for conducting the data collection and the user study and have done the CNIL declaration for this data collection. Our data collection and user study will start early 2015.

RAP Project-Team (section vide)

SOCRATE Project-Team

5. New Software and Platforms

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), proposed by JM Gorce in 2001 and further extended. The discrete ParFlow equations are translated in the Fourier domain providing a large 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 requests. However, we have shown that the use of a multi-resolution approach allows the main computational load to be restricted to a pre-processing phase. Extensive works have been done to make predictions more realistic. The development of the wiplan software has been a part of the european project iPlan (IAPP-FP7 project) and has been integrated in NS-3 simulator.

5.3. FloPoCo

The purpose of the open-source FloPoCo project is to explore the many ways in which the flexibility of the FPGA target can be exploited in the arithmetic realm. FloPoCo is a generator of operators written in C++ and outputting synthesizable VHDL automatically pipelined to an arbitrary frequency. Among the known users of FloPoCo are U. Bristol,U. Cape Town, U.T. Cluj-Napoca, Imperial College, U. Essex, U. Madrid, U. P. Milano, T.U. Muenchen, T. U. Kaiserslautern, U. Paderborn, CalTech, U. Pernambuco, U. Perpignan, U. Tohoku, U. Tokyo, Virginia Tech U. and several companies.

In 2014, FloPoCo was enhanced with a generator of FIR filters accurate to the last bit [19] and several variants of the Atan2 function [46].

Web page: http://flopoco.gforge.inria.fr/

5.4. FIT/CorteXlab software

During the setting up of the FIT/CorteXlab platform, important software tools have been developped for the platform. The main tools is Minus which is used to deploy software programs on SDR hardware, it is developped in Python and is able to deploy complete configuration of NI USRP or Nutaq PicoSDR platforms. A seconf tools is DAS (*Automatic deployement system*) which is used to create the complete software environment of the servers of FIT/CorteXlab. This software could be use to create another testbed based on the same principle: hardware SDR nodes programmed from internet. These softwarea are currently used on the deployement testbed and on the production testbed.

URBANET Team

5. New Software and Platforms

5.1. WSNet

Participants: Rodrigue Domga Komguem, Quentin Lampin, Trista Lin, Alexandre Mouradian, Fabrice Valois (contact).

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

Participants:Marco Fiore (contact), Diala Naboulsi, Razvan Stanica.

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/ .

5.3. PrivaMovApp

Participants:Djamel Benferhat, Patrice Raveneau, Hervé Rivano, Razvan Stanica (contact).

UrbaNet is leading the development of an Android application for user data collection purposes. The application is based on the Funf (http://www.funf.org/) framework, and is currently available on Google Play. A first deployment of the application, on 25 users, took place in December, at the ACM Middleware 2014 conference, in Bordeaux.

5.4. Sense in the City

Participants:Khaled Boussetta (contact), Hervé Rivano, Hamadoun Tall.

We are developing a lightweight experimentation platform for wireless sensor networks. The main objective of this platform is to be easily transferable and deployable on the field. It allows a simplified deployment of the code running on the sensors and the collection of logs generated by the instrumentation of the code on a centralized database. In the early stage of the platform, the sensors are powered by small PCs, e.g. Raspberry Pis, but we are investigating the integration of energy harvesting capabilities such as solar panels. First practical deployments of the platform will be used to showcase some protocols developed in the team in 2015.