Activity Report 2012

Project-Team ARLES

Software architectures and distributed systems

RESEARCH CENTER
Paris - Rocquencourt

THEME
Distributed Systems and Services

Table of contents

Project-Team ARLES

Keywords: Distributed System, Middleware Architectures, Software Engineering, Pervasive Computing, Service Oriented Architecture

Creation of the Project-Team: February 01, 2002 .

1. Members

Research Scientists

Valérie Issarny [Team Leader, Inria Senior Researcher, HdR]
Nikolaos Georgantas [Inria Junior Researcher]
Animesh Pathak [Inria Junior Researcher]
Nelly Bencomo [Marie Curie Fellow]

Engineers

Sandrine Beauche [*Ingénieur Expert*, Inria, until September 2012]
George Bouloukakis [*Ingénieur Expert*, Inria, since September 2012]
Luca Cavallaro [*Ingénieur Expert*, Inria, until March 2012]
Yesid Jarma Alvis [*Ingénieur Expert*, Inria, since May 2012]
George Mathioudakis [*Ingénieur Expert*, Inria, since November 2012]
Bachir Moussa Tari Bako [*Ingénieur Jeune Diplomé*, Inria]
Cong Kinh Nguyen [*Ingénieur Expert*, Inria]
George Rosca [*Ingénieur Jeune Diplomé*, Inria, since October 2012]
Daniel Sykes [*Ingénieur Expert*, Inria, until July 2012]

PhD Students

Emil Andriescu [Inria-Ambientic CIFRE, University Pierre et Marie Curie - Paris 6]
Dionysis Athanasopoulos [Inria, University of Ioannina, Greece]
Amel Bennaceur [Inria, University Pierre et Marie Curie - Paris 6]
Benjamin Billet [Inria University Pierre et Marie Curie - Paris 6]
Sara Hachem [Inria, University of Versailles Saint-Quentin-en-Yvelines]
Pankesh Patel [Inria CORDI-S, University Pierre et Marie Curie - Paris 6]

Post-Doctoral Fellows

Oleg Davidyuk [ERCIM Fellow, since July 2012]
Ajay Kattepur [Inria, since October 2012]

Visiting Scientists

Peter Sawyer [Lancaster University, since November 2012]
Romina Torres [Universidad Técnica Federico, Santa Maria, Valparaíso, Chile, April-July 2012]

Administrative Assistants

Stéphanie Chaix [until September 2012]
Emmanuelle Grousset [October 2012]
Florence Barbara [since November 2012]

2. Overall Objectives

2.1. Overall Objectives

With digital equipment becoming increasingly networked, either on wired or wireless networks, for personal and professional use alike, distributed software systems have become a crucial element in information and communication technologies. The study of these systems forms the core of the ARLES project-team's work, which is specifically concerned with defining new system software architectures, based on the use of emerging networking technologies.

Since the 90s, middleware has emerged as a prominent solution to overcome the heterogeneity of distributed infrastructure. It establishes a software layer that homogenizes infrastructure diversities by means of a well-defined and structured distributed programming model. Moreover, middleware provides building blocks to be exploited by applications for enforcing *non-functional* properties, such as dependability and performance. Finally, by providing reusable solutions for the development of distributed systems, which is increasingly demanding, the role of middleware has proved central in the software system development practice.

However, the development of middleware itself remains a complex task [1]. In particular, middleware must provide the adequate networking and computing abstractions to match the distributed application requirements. Further, while the development of legacy middleware has been significantly driven by requirements of distributed information systems, the ongoing evolution of the networking environment leads to a much broader application of distributed computing — including, among others, the proliferation of disparate mobile computing devices and smart phones, and the inclusion of wireless sensor networks into modern and future pervasive systems. As a result of the above, new requirements arise for middleware, e.g., supporting open and mobile networking, as well as context awareness [5]. Among these new requirements, we are especially interested in applications involving social interactions between the users of modern smart phones, which involve addressing issues of trust and privacy along with efficiency of algorithms to deal with the lack of resources.

In the above context, ARLES studies the engineering of middleware-based systems, with a special emphasis on enabling interoperability in the ubiquitous computing [2] and subsequent pervasive computing [3] and ambient intelligence [4] visions, keeping in view the advances in the constitution of these systems, both in terms of the various communication and discovery protocols used, as well as the hardware and software platforms available. Our research is more specifically focused on eliciting middleware for pervasive computing, from foundations and architectural design to prototype implementations. Proposed middleware and programming abstractions shall then effectively leverage networked resources, in particular accounting for advanced wireless networking technologies, while also addressing concerns raised due to the presence of large numbers of extremely low-power devices such as wireless sensors and actuators. This raises a number of complementary research challenges:

- **System architecture:** How to architect and further program pervasive computing systems out of the resources available in the highly dynamic networking environment?
- **System modeling:** How to abstract and further model the networked resources and related networking environment for distributed pervasive computing?
- **Interoperability:** How to actually overcome the heterogeneity of the pervasive computing environment, including middleware heterogeneity?
- **Networking:** How to benefit from the rich wireless networking technologies?
- **Quality of service:** How to effectively master the high dynamics and openness of pervasive computing environments and, in particular, how to ensure dependability and performance in such environments?

All the above mentioned challenges are inter-related, calling for their study in both the software engineering and distributed systems domains. Indeed, proposed middleware abstractions and related programming models shall effectively foster the development of *robust* distributed software systems, which, at the same time, must be implemented in an *efficient* way. Specifically:

- In the **software engineering domain**, ARLES studies resource abstractions and interaction paradigms to be offered by middleware, together with the associated languages, methods and tools for describing and composing the abstracted resources into applications. Our primary goal is to foster the development of *robust* and *interoperable* distributed systems that are highly dynamic to adapt

[1] V. Issarny *et al*. Systematic Aid for Developing Middleware Architectures. In Communications of the ACM, Special Issue on Adaptive Middleware, 45(6). 2002.

[2] M. Weiser. The Computer for the Twenty-First Century. In Scientific American. 1991.

[3] M. Satyanarayanan. Pervasive Computing: Vision and Challenges. In IEEE Personal Communications. August 2001.

[4] E. Aarts *et al.* Ambient Intelligence. In The Invisible Future: the Seamless Integration of Technology into Everyday Life. McGraw-Hill. 2002.

to the ever-changing networking environment, and further meet quality of service requirements.

- In the **distributed systems domain**, ARLES studies innovative middleware architectures and related distributed algorithms and protocols for the *efficient* networking of distributed resources into distributed pervasive systems, in particular taking into account the high mobility and heterogeneity of the constituent nodes.

2.2. Highlights of the Year

During this year, while we have pursued our research on advanced service-oriented architectures and related middleware solutions for next generation networking environments, we have also made advances over our initial progress in research on several new subjects, called for by the ongoing drastic evolution of the networking environment:

- Dynamic interoperability among networked systems towards making them eternal, by way of on-the-fly generation of connectors based on adequate system models. This research is part of a major European collaborative project within the Future and Emerging Technology program of the EC FP7-ICT (§ 6.2, § 7.2.1.1), which was successfully completed in November 2012 with the highest rating of "Excellent progress (the project has fully achieved its objectives and technical goals for the period and has even exceeded expectations)".
- Interaction paradigm abstractions and service oriented middleware for choreographies in the ultra-large scale future Internet. This research is also part of a major European collaborative project within the Software and Service Architectures and Infrastructures programme of the EC FP7-ICT (§ 6.4, § 7.2.1.2).

3. Scientific Foundations

3.1. Introduction

Research undertaken within the ARLES project-team aims to offer comprehensive solutions to support the development of pervasive computing systems that are dynamically composed according to networked resources in the environment. This leads us to investigate methods and tools supporting the engineering of pervasive software systems, with a special emphasis on associated middleware solutions.

3.2. Engineering Pervasive Software Systems

Since its emergence, middleware has proved successful in assisting distributed software development, making development faster and easier, and significantly promoting software reuse while overcoming the heterogeneity of the distributed infrastructure. As a result, middleware-based software engineering is central to the principled development of pervasive computing systems. In this section, we (i) discuss challenges that middleware brings to software engineering, and (ii) outline a revolutionary approach to middleware-based software engineering aiming at the dynamic runtime synthesis of emergent middleware.

3.2.1. Middleware-based Software Engineering

Middleware establishes a new software layer that homogenizes the infrastructure's diversities by means of a well-defined and structured distributed programming model, relieving software developers from low-level implementation details, by: (i) at least abstracting transport layer network programming via high-level network abstractions matching the application computational model, and (ii) possibly managing networked resources to offer quality of service guarantees and/or domain specific functionalities, through reusable middleware-level services. More specifically, middleware defines:

- A resource definition language that is used for specifying data types and interfaces of networked software resources;

- A high-level addressing scheme based on the underlying network addressing scheme for locating resources;
- Interaction paradigms and semantics for achieving coordination;
- A transport/session protocol for achieving communication; and
- A naming/discovery protocol with related registry structure and matching relation for publishing and discovering the resources available in the given network.

Attractive features of middleware have made it a powerful tool in the software system development practice. Hence, middleware is a key factor that has been and needs to be further taken into account in the Software Engineering (SE) discipline [5]. The advent of middleware standards have further contributed to the systematic adoption of this paradigm for distributed software development.

In spite of the above, mature engineering methodologies to comprehensively assist the development of middleware-based software systems, from requirements analysis to deployment and maintenance, are lagging behind. Indeed, systematic software development accounting for middleware support is rather the exception than the norm, and methods and related tools are dearly required for middleware-based software engineering. This need becomes even more demanding if we consider the diversity and scale of today's networking environments and application domains, which makes middleware and its association with applications highly complex [5], raising new, challenging requirements for middleware. Among those, access to computational resources should be open across network boundaries and dynamic due to the potential mobility of host- and user-nodes. This urges middleware to support methods and mechanisms for description, dynamic discovery and association, late binding, and loose coordination of resources. In such variable and unpredictable environments, operating not only according to explicit system inputs but also according to the context of system operation becomes of major importance, which should be enabled by the middleware. Additionally, the networking infrastructure is continuing to evolve at a fast pace, and suggesting new development paradigms for distributed systems, calling for next-generation middleware platforms and novel software engineering processes integrating middleware features in all phases of the software development.

3.2.2. Beyond Middleware-based Architectures for Interoperability

As discussed above, middleware stands as the conceptual paradigm to effectively network together heterogeneous systems, specifically providing upper layer interoperability. That said, middleware is yet another technological block, which creates islands of networked systems.

Interoperable middleware has been introduced to overcome middleware heterogeneity. However, solutions remain rather static, requiring either use of a proprietary interface or a priori implementation of protocol translators. In general, interoperability solutions solve protocol mismatch among middleware at syntactic level, which is too restrictive. This is even truer when one considers the many dimensions of heterogeneity, including software, hardware and networks, which are currently present in ubiquitous networking environments, and that require fine tuning of the middleware according to the specific capacities embedded within the interacting parties. Thus, interoperable middleware can at best solve protocol mismatches arising among middleware aimed at a specific domain. Indeed, it is not possible to a priori design a universal middleware solution that will enable effective networking of digital systems, while spanning the many dimensions of heterogeneity currently present in networked environments and further expected to increase dramatically in the future.

A revolutionary approach to the seamless networking of digital systems is to synthesize connectors on the fly, via which networked systems communicate. The resulting emergent connectors then compose and further adapt the interaction protocols run by the connected systems, from the application layer down to the middleware layer. Hence, thanks to results in this new area, networked digital systems will survive the obsolescence of interaction protocols and further emergence of new ones.

We have specifically undertaken cooperative research on the dynamic synthesis of emergent connectors which shall rely on a formal foundation for connectors that allows learning, reasoning about, and adapting the interaction behavior of networked systems [6]. Further, compared to the state of the art foundations for

[5] W. Emmerich. Software Engineering and Middleware: a roadmap. In Proceedings of the Conference on the Future of Software Engineering, Limerick, Ireland, Jun. 2000.

connectors, it should operate a drastic shift by learning, reasoning about, and synthesizing connector behavior at run-time. Indeed, the use of connector specifications pioneered by the software architecture research field has mainly been considered as a design-time concern, for which automated reasoning is now getting practical even if limitations remain. On the other hand, recent effort in the semantic Web domain brings ontology-based semantic knowledge and reasoning at run-time; however, networked system solutions based thereupon are currently mainly focused on the functional behavior of networked systems, with few attempts to capture their interaction behavior as well as non-functional properties. In this new approach, the interaction protocols (both application- and middleware-layer) behavior will be learnt by observing the interactions of the networked systems, where ontology-based specification and other semantic knowledge will be exploited for generating connectors on the fly.

3.3. Middleware Architectures for Pervasive Computing

Today's wireless networks enable dynamically setting up temporary networks among mobile nodes for the realization of some distributed function. However, this requires adequate development support and, in particular, supporting middleware platforms for alleviating the complexity associated with the management of dynamic networks composed of highly heterogeneous nodes. In this section, we present an overview of: (i) service oriented middleware, a prominent paradigm in large distributed systems today, and (ii) middleware for wireless sensor networks, which have recently emerged as a promising platform.

3.3.1. Service Oriented Middleware

The *Service Oriented Computing*(SOC) paradigm advocates that networked resources should be abstracted as services, thus allowing their open and dynamic discovery, access and composition, and hence reuse. Due to this flexibility, SOC has proven to be a key enabler for pervasive computing. Moreover, SOC enables integrating pervasive environments into broader service oriented settings: the current and especially the *Future Internet* is the ultimate case of such integration. We, more particularly, envision the Future Internet as a ubiquitous setting where services representing resources, people and things can be freely and dynamically composed in a decentralized fashion, which is designated by the notion of service choreography in the SOC idiom. In the following, we discuss the role that *service oriented middleware* is aimed to have within our above sketched vision of the Future Internet, of which pervasive computing forms an integral part.

From service oriented computing to service oriented middleware: In the last few years, there is a growing interest in choreography as a key concept in forming complex service-oriented systems. Choreography is put forward as a generic abstraction of any possible collaboration among multiple services, and integrates previously established views on service composition, among which service orchestration. Several different approaches to choreography modeling can be found in the literature: *Interaction-oriented* models describe choreography as a set of interactions between participants; while *process-oriented* models describe choreography as a parallel composition of the participants' business processes. *Activity-based* models focus on the interactions between the parties and their ordering, whereas the state of the interaction is not explicitly modeled or only partly modeled using variables; while *state-based* models model the states of the choreography as first-class entities, and the interactions as transitions between states.

The above modeling categorizations are applied in the ways in which: service choreographies are specified (e.g., by employing languages such as BPMN, WS-CDL, BPEL); services are discovered, selected and composed into choreographies (e.g., based on their features concerning interfaces, behavior, and non-functional properties such as QoS and context); heterogeneity between choreographed services is resolved via adaptation (e.g., in terms of service features and also underlying communication protocols); choreographies are deployed and enacted (e.g., in terms of deployment styles and execution engines); and choreographies are maintained/adapted given the independent evolution of choreographed services (e.g., in terms of availability

[6] Valérie Issarny, Bernhard Steffen, Bengt Jonsson, Gordon S. Blair, Paul Grace, Marta Z. Kwiatkowska, Radu Calinescu, Paola Inverardi, Massimo Tivoli, Antonia Bertolino, Antonino Sabetta: CONNECT Challenges: Towards Emergent Connectors for Eternal Networked Systems. In Proceedings of ICECCS 2009.

and QoS). These are demanding functionalities that service oriented middleware should provide for supporting service choreographies. In providing these functionalities in the context of the Future Internet, service oriented middleware is further challenged by two key Future Internet properties: its *ultra large scale* as in number of users and services, and the *high degree of heterogeneity* of services, whose hosting platforms may range from that of resource-rich, fixed hosts to wireless, resource-constrained devices. These two properties call for considerable advances to the state of the art of the SOC paradigm.

Our work in the last years has focused on providing solutions to the above identified challenges, more particularly in the domain of pervasive computing. Given the prevalence of mobile networking environments and powerful hand-held consumer devices, we consider resource constrained devices (and things, although we focus on smart, i.e., computation-enabled, things) as first-class entities of the Future Internet. Concerning middleware that enables networking mobile and/or resource constrained devices in pervasive computing environments, several promising solutions have been proposed, such as mobile Gaia, TOTA, AlfredO, or work at UCL, Carnegie Mellon University, and the University of Texas at Arlington. They address issues such as resource discovery, resource access, adaptation, context awareness as in location sensitivity, and pro-activeness in a seamless manner. Other solutions specialize in sensor networks; we, more specifically, discuss middleware for wireless sensor networks in the next section. In this very active domain of service-oriented middleware for pervasive computing environments, we have extensive expertise that ranges from lower-level cross-layer networking to higher-level semantics of services, as well as transversal concerns such as context and privacy. We have in particular worked on aspects including semantic discovery and composition of services based on their functional properties, heterogeneity of service discovery protocols, and heterogeneity of network interfaces. Based on our accumulated experience, we are currently focusing on some of the still unsolved challenges identified above.

QoS-aware service composition: With regard to service composition in pervasive environments, taking into account QoS besides functional properties ensures a satisfactory experience to the end user. We focus here on the orchestration-driven case, where service composition is performed to fulfill a task requested by the user along with certain QoS constraints. Assuming the availability of multiple resources in service environments, a large number of services can be found for realizing every sub-task part of a complex task. A specific issue emerges in this regard, which is about selecting the best set of services (i.e., in terms of QoS) to participate in the composition, meeting user's global QoS requirements. QoS-aware composition becomes even more challenging when it is considered in the context of dynamic service environments characterized by changing conditions. As dynamic environments call for fulfilling user requests on the fly (i.e., at run-time) and as services' availability cannot be known a priori, service selection and composition must be performed at runtime. Hence, the execution time of service selection algorithms is heavily constrained, whereas the computational complexity of this problem is NP-hard.

Coordination of heterogeneous distributed systems: Another aspect that we consider important in service composition is enabling integration of services that employ different interaction paradigms. Diversity and ultra large scale of the Future Internet have a direct impact on coordination among interacting entities. Our choice of choreography as global coordination style among services should further be underpinned by support for and interoperability between heterogeneous interaction paradigms, such as message-driven, event-driven and data-driven ones. Different interaction paradigms apply to different needs: for instance, asynchronous, event-based publish/subscribe is more appropriate for highly dynamic environments with frequent disconnections of involved entities. Enabling interoperability between such paradigms is imperative in the extremely heterogeneous Future Internet integrating services, people and things. Interoperability efforts are traditionally based on, e.g., bridging communication protocols, where the dominant position is held by ESBs, wrapping systems behind standard technology interfaces, and/or providing common API abstractions. However, such efforts mostly concern a single interaction paradigm and thus do not or only poorly address cross-paradigm interoperability. Efforts combining diverse interaction paradigms include: implementing the LIME tuple space middleware on top of a publish/subscribe substrate; enabling Web services/SOAP-based interactions over a tuple space binding; and providing ESB implementations based on the tuple space paradigm.

Evolution of service oriented applications: A third issue we are interested in concerns the maintenance of service-oriented applications despite the evolution of employed services. Services are autonomous systems that have been developed independently from each other. Moreover, dynamics of pervasive environments and the Future Internet result in services evolving independently; a service may be deployed, or un-deployed at anytime; its implementation, along with its interface may change without prior notification. In addition, there are many evolving services that offer the same functionality via different interfaces and with varying quality characteristics (e.g., performance, availability, reliability). The overall maintenance process amounts to replacing a service that no longer satisfies the requirements of the employing application with a substitute service that offers the same or a similar functionality. The goal of seamless service substitution is to relate the substitute service with the original service via concrete mappings between their operations, their inputs and outputs. Based on such mappings, it is possible to develop/generate an adapter that allows the employing application to access the substitute service without any modification in its implementation. The service substitution should be dynamic and efficient, supported by a high level of automation. The state of the art in service substitution comprises various approaches. There exist efforts, which assume that the mappings between the original and the substitute service are given, specified by the application or the service providers. The human effort required makes these approaches impractical, especially in the case of pervasive environments. On the other hand, there exist automated solutions, proposing mechanisms for the derivation of mappings. The complexity of these approaches scales up with the cardinality of available services and therefore efficiency is compromised. Again, this is an important disadvantage, especially considering the case of pervasive environments.

3.3.2. Middleware for Wireless Sensor Networks

Wireless sensor networks (WSNs) enable low cost, dense monitoring of the physical environment through collaborative computation and communication in a network of autonomous sensor nodes, and are an area of active research. Owing to the work done on system-level functionalities such as energy-efficient medium access and data-propagation techniques, sensor networks are being deployed in the real world, with an accompanied increase in network sizes, amount of data handled, and the variety of applications. The early networked sensor systems were programmed by the scientists who designed their hardware, much like the early computers. However, the intended developer of sensor network applications is not the computer scientist, but the designer of the system *using* the sensor networks, which might be deployed in a building or a highway. We use the term *domain expert* to mean the class of individuals most likely to use WSNs – people who may have basic programming skills but lack the training required to program distributed systems. Examples of domain experts include architects, civil and environmental engineers, traffic system engineers, medical system designers etc. We believe that the wide acceptance of networked sensing is dependent on the ease-of-use experienced by the domain expert in developing applications on such systems.

The obvious solution to enable this ease-of-use in application development is sensor network middleware, along with related programming abstractions [7]. Recent efforts in standardizing network-layer protocols for embedded devices provide a sound foundation for research and development of middleware that assist the sensor network developers in various aspects that are of interest to us, including the following.

Data-oriented operations: A large number of WSN applications are concerned with sampling and collection of data, and this has led to a large body of work to provide middleware support to the programmer of WSNs for easy access to the data generated and needed by the constituent nodes. Initial work included Hood, and TeenyLIME, which allowed data-sharing over a limited spatial range. Further work proposed the use of the DART runtime environment, which exposes the sensor network as a distributed data-store, addressable by using logical addresses such as “all nodes with temperature sensors in Room 503”, or “all fire sprinklers in the fifth and sixth floors”, which are more intuitive than, say, IP addresses. Taking a different approach toward handling the data in the sensor network, some middleware solutions propose to manipulate them using semantic techniques, such as in the Triple Space Computing approach, which models the data shared by the nodes in the system as RDF triples (subject-predicate-object groups), a standard method for semantic data

[7] L. Mottola and G. P. Picco. Programming Wireless Sensor Networks: Fundamental Concepts and State of the Art. In ACM Computing Surveys. Volume 43, Issue 3. April 2011.

representation. They propose to make these triples available to the participating nodes using a tuple space, thus giving it the "triple space" moniker. S-APL or Semantic-Agent Programming Language uses semantic technologies to integrate the semantic descriptions of the domain resources with the semantic prescription of agent behavior.

Integration with non-WSN nodes: Most of the work above focuses on designing applications that exhibit only intra-network interactions, where the interaction with the outside world is only in the form of sensing it, or controlling it by actuation. The act of connecting this data to other systems outside the sensor network is mostly done using an external gateway. This is then supported by middlewares that expose the sensor network as a database (e.g., TinyDB and Cougar), allowing the operator to access the data using a SQL-like syntax, augmented with keywords that can be used to specify the rate of sampling, for example. Another direction of integrating WSNs in general with larger systems such as Web servers has been toward using REST (REpresentational State Transfer) technologies, which are already used for accessing services on the Web as a lightweight alternative to SOAP. There has also been work proposing a system that will enable heterogeneous sensor and actuators to expose their sensing and actuation capabilities in a plug and play fashion. It proposes a middleware that defines a set of constraints, support services and interaction patterns that follow the REST architectural style principles, using the ATOM Web publishing protocol for service description, and a two-step discovery process. Additionally, there has been work in implementation of a REST-oriented middleware that runs on embedded devices such as Sun SPOT nodes, and the Plogg wireless energy monitors. This involves a two-fold approach — embedding tiny Web servers in devices that can host them, and employing a proxy server in situations where that is not the case. However, it has been noticed that the abstractions provided by REST might be too simplistic to compose complex applications over the services provided by WSN nodes. Some of the most recent work in this area also proposes to convert existing (network-layer) gateways into smart gateways, by running application code on them.

In addition to supporting the above interactions, sensor network middleware has also been proposed to address the challenges arising from the fact that a particular sensor or actuator may not be always available. This leads to the need for transparent reconfiguration, where the application developer should not have to care about reliability issues. The PIRATES event-based middleware for resource-rich nodes (hosting sensors/actuators, or just processing data) includes a third-party-remapping facility that can be used to remap a component's endpoints without affecting the business logic. In that sense, it is similar to the RUNES middleware targeted at embedded systems.

Finally, we also note the recent initial WSN middleware research focused on the new nascent classes of systems. Most recently, the field of *participatory sensing* [8] has emerged, where the role of sensing is increasingly being performed by the mobile phones carried by the users of the system, providing data captured using the sound, GPS, accelerometer and other sensors attached to them. This has led to the emergence of middleware such as JigSaw. The core additional challenges in this domain come from the inherent mobility of the nodes, as well as their extremely large scale.

4. Application Domains

4.1. Application Domains

The ARLES project-team is interested in the application of pervasive computing, and as such considers various application domains. Indeed, our application domain is voluntarily broad since we aim at offering generic solutions. However, we examine exploitation of our results for specific applications, as part of the experiments that we undertake to validate our research results through prototype implementation. Applications that we consider in particular include demonstrators developed in the context of the European and National projects to which we contribute (§ 7.1 & 7.2).

[8] Lane, N.D.; Miluzzo, E.; Hong Lu; Peebles, D.; Choudhury, T.; Campbell, A.T.; , "A survey of mobile phone sensing," Communications Magazine, IEEE , vol.48, no.9, pp.140-150, Sept. 2010

5. Software

5.1. Introduction

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

5.2. iCONNECT – Emergent Middleware Enablers

Participant: Valérie Issarny [correspondent].

As part of our research work on Emergent Middleware, we have implemented Enablers (or Enabler functionalities) that make part of the overall CONNECT architecture realizing Emergent Middleware in practice [2]. The focus of ARLES work is on the: *Discovery enabler* that builds on our extensive background in the area of interoperable pervasive service discovery; and *Synthesis enabler* that synthesizes mediators that allow networked systems that have compatible functionalities to interact despite mismatching interfaces and/or behaviors.

The CONNECT Discovery Enabler is the component of the overall CONNECT architecture that handles discovery of networked systems (NSs), stores their descriptions (NS models), and performs an initial phase of matchmaking to determine which pairs of systems are likely to be able to interoperate. Such pairs are then passed to the Synthesis Enabler so that mediators can be generated. The Discovery Enabler is written in Java and implements several legacy discovery protocols including DPWS and UPnP.

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

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

5.3. Service-oriented Middleware for Pervasive Computing

Participants: Nikolaos Georgantas [correspondent], Valérie Issarny [correspondent].

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

- **WSAMI - A Middleware Based on Web Services for Ambient Intelligence:** WSAMI (Web Services for AMbient Intelligence) is based on the Web services architecture and allows for the deployment of services on wireless handheld devices like smartphones and PDAs.
  URL: http://www-rocq.inria.fr/arles/download/ozone/index.htm
- **Ariadne - A Protocol for Scalable Service Discovery in MANETs:** Ariadne enriches WSAMI with the Ariadne service discovery protocol, which has been designed to support decentralized Web service discovery in multi-hop mobile ad hoc networks (MANETs). Ariadne enables small and resource-constrained mobile devices to seek and find complementary, possibly mobile, Web services needed to complete specified tasks in MANETs, while minimizing the traffic generated and tolerating intermittent connectivity.

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

- **MUSDAC - A Middleware for Service Discovery and Access in Pervasive Networks:** The MUlti-protocol Service Discovery and ACcess (MUSDAC) middleware platform enriches WSAMI so as to enable the discovery and access to services in the pervasive environment, which is viewed as a loose and dynamic composition of independent networks. MUSDAC manages the efficient dissemination of discovery requests between the different networks and relies on specific plug-ins to interact with the various middleware used by the networked services.
  URL: http://www-rocq.inria.fr/arles/download/ubisec/index.html
- **INMIDIO - An Interoperable Middleware for Ambient Intelligence:** INMIDIO (INteroperable MIddleware for service Discovery and service InteractiOn) dynamically resolves middleware mismatch. More particularly, INMIDIO identifies the interaction middleware and also the discovery protocols that execute on the network and translates the incoming/outgoing messages of one protocol into messages of another, target protocol.
  URL: http://www-rocq.inria.fr/arles/download/inmidio/index.html
- **COCOA - A Semantic Service Middleware:** COCOA is a comprehensive approach to semantic service description, discovery, composition, adaptation and execution, which enables the integration of heterogeneous services of the pervasive environment into complex user tasks based on their abstract specification. Using COCOA, abstract user tasks are realized by dynamically composing the capabilities of services that are currently available in the environment.
  URL: http://gforge.inria.fr/projects/amigo/
- **ubiSOAP - A Service Oriented Middleware for Seamless Networking:** ubiSOAP brings multi-radio, multi-network connectivity to services through a comprehensive layered architecture: (i) the multi-radio device management and networking layers together abstract multi-radio connectivity, selecting the optimal communication link to/from nodes, according to quality parameters; (ii) the communication layer allows for SOAP-based point-to-point and group-based interactions in the pervasive network; and (iii) the middleware services layer brings advanced distributed resource management functionalities customized for the pervasive networking environment.
  URL: http://www.ist-plastic.org.

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

Participant: Nikolaos Georgantas [correspondent].

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

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

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

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

5.5. Srijan: Data-driven Macroprogramming for Sensor Networks

Participant: Animesh Pathak [correspondent].

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

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

1. Specification of application as a task graph,
2. Customization of the auto-generated source files with domain-specific imperative code,
3. Specification of the target system structure,
4. Compilation of the macroprogram into individual customized runtimes for each constituent node of the target system, and finally
5. Deployment of the auto generated node-level code in an over-the-air manner to the nodes in the target system.

The current implementation of *Srijan* targets both the Sun SPOT sensor nodes and larger nodes with J2SE. Most recently, *Srijan* also includes rudimentary support for incorporating Web services in the application being designed. The software is released under open source license, and available as an Eclipse plug-in at http://code.google.com/p/srijan-toolkit/.

5.6. Yarta: Middleware for supporting Mobile Social Applications

Participant: Animesh Pathak [correspondent].

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

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

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

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

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

Participant: Valérie Issarny [correspondent].

Building on the lessons learned with the development of pervasive service oriented middleware and of applications using them, we have been developing the custom iBICOOP middleware. iBICOOP specifically aims at assisting the development of advanced mobile, collaborative application services by supporting interactions between mobile users.

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

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

- The *Communication Manager* provides mechanisms to communicate over different available network interfaces of a device — Bluetooth, WiFi, Cellular — and also using different technologies e.g., Web services, HTTP/TCP sockets, ad hoc mode.
- The *Security Manager* uses well-established techniques of cryptography and secure communication to provide necessary security.
- The *Partnership Manager* provides device or user information in the form of *profiles*.
- iBICOOP relies on service location protocols for *naming and discovery* of nearby services on currently active network interfaces that support IP multicast.
- Besides normal file managing tasks, the *Local File Manager* gives the user clear cues to the files that have been replicated across multiple devices or shared among different users by using different icons.

The iBICOOP middleware has been licensed by AMBIENTIC (http://www.ambientic.com/), a start-up that specifically develops innovative mobile distributed services on top of the iBICOOP middleware that allows for seamless interaction and content sharing in today's multi-* networks.
URL: https://www-roc.inria.fr/arles/index.php/ongoing-research-projects/74-data-sharing-and-replication-in-pervasive-networks

6. New Results

6.1. Introduction

The ARLES project-team investigates solutions in the forms of languages, methods, tools and supporting middleware to assist the development of distributed software systems, with a special emphasis on mobile distributed systems enabling the ambient intelligence/pervasive computing vision. Our research activities in 2012 have focused on the following areas:

- Dynamic interoperability among networked systems toward making them eternal, by way of on-the-fly generation of connectors based on adequate system models (§ 6.2);
- Pervasive service-oriented software engineering, focusing on supporting service composition in an increasingly heterogeneous and dynamic networking environment, while enforcing quality of service (§ 6.3);
- Service oriented middleware for the ultra large scale future Internet of Things (§ 6.4);
- Abstractions for enabling domain experts to easily compose applications on the Internet of Things (§ 6.5); and
- The use of Requirement Engineering techniques for enabling systems to be self-adaptive under uncertainty (§ 6.6).

6.2. Emergent Middleware Supporting Interoperability in Extreme Distributed Systems

Participants: Emil Andriescu, Amel Bennaceur, Luca Cavallaro, Valérie Issarny, Daniel Sykes.

Interoperability is a fundamental challenge for today's extreme distributed systems. Indeed, the high-level of heterogeneity in both the application layer and the underlying infrastructure, together with the conflicting assumptions that each system makes about its execution environment hinder the successful interoperation of independently developed systems. A wide range of approaches have thus been proposed to address the interoperability challenge. However, solutions that require performing changes to the systems are usually not feasible since the systems to be integrated may be legacy systems, COTS (Commercial Off-The-Shelf) components or built by third parties; neither are the approaches that prune the behavior leading to mismatches since they also restrict the systems' functionality. Therefore, many solutions that aggregate the disparate systems in a non-intrusive way have been investigated. These solutions use intermediary software entities, called *mediators*, to interconnect systems despite disparities in their data and/or interaction models by performing the necessary coordination and translations while keeping them loosely-coupled. However, creating mediators requires a substantial development effort and a thorough knowledge of the application-domain, which is best understood by domain experts. Moreover, the increasing complexity of today's distributed systems, sometimes referred to as Systems of Systems, makes it almost impossible to develop 'correct' mediators manually. Therefore, formal approaches are used to synthesize mediators automatically.

In light of the above, we have introduced the notion of *emergent middleware* for realizing mediators. Our research on enabling emergent mediators is done in collaboration with our partners of the CONNECT project (§ 7.2.1.1). Our work during the year has more specifically focused on:

- **Architecture enabling emergent middleware.** We have been finalizing, together with our partners in the CONNECT project, the definition of an overall distributed system architecture supporting emergent middleware, from the discovery of networked systems to the learning of their respective behavior and synthesis of emergent middleware enabling them to interoperate [31].
- **Affordance inference.** We have proposed an ontology-based formal model of networked systems based on their affordances (high-level functionalities), interfaces, behavior, and non-functional properties, each of which describes a different facet of the system in a way similar to the service description promoted for semantic Web services. However, legacy systems do not necessarily specify all of the aforementioned facets. Therefore, we have explored techniques to infer the affordance by using textual descriptions of the interface of networked systems. More specifically, we rely on machine learning techniques to automate the inference of the affordance from the interface description by classifying the natural-language text according to a predefined ontology of affordances. In a complementary way, CONNECT partners investigate protocol-learning algorithms to learn the behavior of networked systems on the fly [17].

- **Mediator synthesis for emergent middleware.** We focus on systems that have compatible functionalities, i.e., semantically matching affordances, but are unable to interact successfully due to mismatching interfaces or behaviors. To solve such mismatches, we propose a *mapping-based* approach, whose goal is to automatically synthesize a mediator model that ensures the *safe* interaction of functionnally compatible systems, i.e., deadlock-freedom and the absence of unspecified receptions. Our approach combines semantic reasoning and constraint programming to identify the semantic correspondence between networked systems' interfaces, i.e., *interface mapping*. Unlike existing approaches that only tackle the one-to-one correspondence between actions and for which we investigated a solution using ontology-based model checking [16], the proposed mapping-based approach handles the more general cases of one-to-many and many-to-many mappings. This work has resulted in a supporting software prototype that allows validating the approach; related publication is under writing. A further key research issue we are addressing in emergent middleware is the study of cross-paradigm interaction so as to enable interoperability among highly heterogeneous services (e.g., an IT-based service will likely interact using the client-service scheme while thing-based services rather adopt asynchronous protocols). Toward that goal, we are studying abstract models associated with popular interaction paradigms and higher level, generic interaction paradigms to define cross-paradigm mappings that respect the behavioral semantics of the interacting systems.
- **Automated mediation for cross-layer protocol interoperability.** While existing approaches to interoperability consider either application or middleware heterogeneity separately, we believe that in real world scenarios this does not suffice: application and middleware boundaries are ill-defined and solutions to interoperability must consider them in conjunction. As part of our recent work, we have proposed such a solution, which solves cross-layer interoperability by automatically generating parsers and composers that abstract physical message encapsulation layers into logical protocol layers, thus supporting application layer mediation. Specifically, we support the automated synthesis of mediators at the application layer using the mapping-based approach discussed above, while we introduce *Composite Cross- Layer (CCL) parsers and composers* to handle cross-layer heterogeneity and to provide an abstract representation of the application data exchanged by the interoperating components. In particular, we associate the data embedded in messages with annotations that refers to concepts in a domain ontology. As a result, we are able to reason about the semantics of messages in terms of the operations and the data they require from or provide to one another and automatically synthesize, whenever possible, the appropriate mediators. We have demonstrated the validity of our approach by using the framework to solve cross-layer interoperability between existing conference management systems.
- **Models@run.time.** We have recently integrated the notion of *Models@run.time* in our research towards emergent middleware. We use Models@run.time to extend the applicability of models and abstractions to the runtime environment. As is the case for software development models, a run-time model is often created to support reasoning. However, in contrast to development models, run-time models are used to reason about the operating environment and runtime behavior, and thus these models must capture abstractions of runtime phenomena. Different dimensions need to be balanced, including resource-efficiency (time, memory, energy), context-dependency (time, location, platform), as well as personalization (quality-of-service specifications, profiles). The hypothesis is that because Models@run.time provide meta-information for these dimensions during execution, run-time decisions can be facilitated and better automated.

  Thus, we anticipate that Models@run.time will play an integral role in the management of extremely distributed systems. Our way of using runtime models captures syntax and also semantics of behaviour and supports runtime reasoning. Prior models@run.time approaches have generally concentrated on architectural-based runtime models and self-adaptation of existing software artifacts. However, such artefacts cannot always be produced in advance, and we believe that models@runtime have a fundamental role to play in the production of dynamic, adaptive, and on-the-fly software as investigated in the context of emergent middleware [8]. Specifically, two important methods underpin our approach: *i*) automatic inference of the required runtime models during execution and their

refinement by exploiting learning and synthesis techniques; and *ii*) using these models for a dynamic software synthesis approach, where mediators are formally characterized (using LTS) to allow the runtime synthesis of software.

In order to enable emergent middleware, we have shown how systems can infer information to build runtime models during execution. Importantly, ontologies were exploited to enrich the runtime models and facilitated the mutual understanding required to perform the matching and mapping between the networked heterogeneous systems. Such reasoning about information that was not necessarily known before execution, is in contrast to the traditional use of models@run.time.

6.3. Revisiting the Abstractions of Service Oriented Computing for the Future Internet

Participants: Dionysis Athanasopoulos, Sandrine Beauche, George Bouloukakis, Oleg Davidyuk, Nikolaos Georgantas, Valérie Issarny, Ajay Kattepur.

A software architecture style characterizes, via a set of abstractions, the types of: components (i.e., units of computation or data stores), connectors (i.e., interaction protocols) and possibly configurations (i.e., system structures) that serve to build a given class of systems. As such, the definition of a software architectural style is central toward eliciting appropriate design, development and runtime support for any family of systems. The service oriented architecture style may then be briefly defined as follows: (1) components map to services, which may be refined into consumer, producer or prosumer services; (2) connectors map to traditional client-service interaction protocols; and (3) configurations map to compositions of services through (service-oriented) connectors, e.g., choreography and orchestration structures. While the service-oriented architecture style is well suited to support the development of Internet-based distributed systems, it is largely challenged by the Future Internet that poses new demands in terms of sustaining *ities* such as scalability, heterogeneity, mobility, awareness & adaptability that come in extreme degrees compared to the current Internet. Therefore, we have been working on eliciting software architectural abstractions for the Future Internet by building upon the service-oriented architecture style, as well as on applying them to system design, development and execution.

Complex distributed applications in the Future Internet will be to a large extent based 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). These heterogeneous system domains differ significantly in terms of interaction paradigms, communication protocols, and data representation models, provided by supporting middleware platforms. Specifically considering interaction paradigms, the client/server (CS), publish/subscribe (PS), and tuple space (TS) paradigms are among the most widely employed ones today, with numerous related middleware platforms. In light of the above, we have aimed at eliciting abstractions that (i) leverage the diversity of interaction paradigms associated with today's and future complex distributed systems, as well as (ii) enable cross-paradigm interaction to sustain interoperability in the highly heterogeneous Future Internet.

Existing cross-domain interoperability efforts are based on bridging communication protocols, wrapping systems behind standard technology interfaces, and/or providing common API abstractions. In particular, such techniques have been applied by the two widely established system integration paradigms, that is, service oriented architecture (SOA) and enterprise service bus (ESB). However, state of the art interoperability efforts do not or only poorly address interaction paradigm interoperability. Indeed, systems integrated via SOA and ESB solutions have their interaction semantics transformed to the CS paradigm. Then, potential loss of interaction semantics can result in suboptimal or even problematic system integration. To overcome the limitation of today's ESB-based connectors for cross-domain interoperability in the Future Internet, we introduce a new connector type, called GA connector, which stands for "Generic Application connector". The proposed connector type is based on the service bus paradigm in that it achieves bridging across heterogeneous connector types. However, the behavior of the GA connector type differs from that of classical ESB connectors

by bridging protocols across heterogeneous paradigms, which is further realized by paying special attention to the preservation of the semantics of the composed protocols. Indeed, the GA connector type is based on the abstraction and semantic-preserving merging of the common high-level semantics of base interaction paradigms.

Eliciting Interaction Paradigm Abstractions: We introduce a systematic abstraction of interaction paradigms with the following features:

- First, we introduce base CS, PS and TS connector types, which formally characterize today's core interaction paradigms. The proposed types comprehensively cover the essential semantics of the considered paradigms, based on a thorough survey of the related literature and representative middleware instances.
- Then, we further abstract these connector types into a single higher-level one, the GA connector type. GA is a comprehensive connector type based on the abstract union of CS, PS, and TS, where precise identification of the commonalities or similarities between the latter has enabled the optimization of the former. Further, GA preserves by construction the semantics of CS, PS, and TS.
- In more detail, connector types are formally specified in terms of: (i) their API (Application Programming Interface), and (ii) their roles, i.e., the semantics of interaction of the connected component(s) with the environment via the connector. Regarding the latter, the behavioral specification of roles from a middleware perspective relates to specifying the production and consumption of information in the network, while the semantics of the information are abstracted and dealt with at the application layer. The behaviors of the connector roles are then specified using Labeled Transition Systems (LTS). We precisely define the mapping of the roles implemented by the base connector types to/from the corresponding roles of the GA connector type.
- For both the above abstraction transformations, we provide counterpart concretizations, which enable transforming GA connector primitives to CS, PS, or TS connector primitives and then to concrete middleware platforms primitives.
- Furthermore, based on the GA abstraction, we introduce mapping transformations between any pair from the set {CS, PS, TS} via GA. The fine knowledge of CS, PS, and TS semantics, as embedded in GA, enables these mappings to be precise: differing semantics are mapped to each other in such a way that loss of semantics is limited to the minimum. These mappings relate to the definition of the glue process implemented by the GA connector, which defines how a pair of producer and consumer roles coordinates in the environment. The GA glue reconciles consumer and producer roles that may differ with respect to time and space coupling as well as scoping. Hence, GA connectors support interactions among highly heterogeneous services of the Future Internet, and especially across domains.

eXtensible Service Bus: We apply the above connector abstractions to introduce an enhanced bus paradigm, the *eXtensible Service Bus (XSB)*. XSB features richer interaction semantics than common ESB implementations 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. When mapping between such paradigms, special attention is paid to the preservation of interaction semantics. XSB has the following features:

- XSB is an abstract bus that prescribes only the high-level semantics of the common bus protocol. The XSB common bus protocol features GA semantics.
- Heterogeneous systems can be plugged into the XSB by employing binding components that adapt between the native middleware of the deployed system and the common bus protocol. This adaptation is based on the systematic abstractions and mappings discussed above
- XSB, being an abstract bus, can have different implementations. This means that it needs to be complemented with a substrate which at least supports: (1) deployment (i.e., plugging) of various systems on the bus, and (2) a common bus protocol implementing GA semantics. With respect to the latter, we envision that a GA protocol realization may either be designed and built from scratch (still

supposing at least an IP-based transport substrate) or be implemented by conveying GA semantics on top of an existing higher-level protocol used as transport carrier. The latter solution can be attractive, as it facilitates GA protocol realizations in different contexts and domains.

We have carried out two realizations of XSB for the CHOReOS project [30], the first on PEtALS ESB and the second on EasyESB. The genericity and modularity of our solution allowed for easily porting from the first implementation to the second one. We support interoperable peer-to-peer interaction among the CS, PS, and TS paradigms and provide templates for systematic and highly facilitated building of binding components for middleware platforms that follow any one of the three paradigms.

6.4. Service Oriented Middleware facing the Challenges of the Internet of Things

Participants: Benjamin Billet, Nikolaos Georgantas, Sara Hachem, Valérie Issarny, Yesid Jarma Alvis, Cong Kinh Nguyen, George Mathioudakis.

In our vision, The Future Internet can be defined as the union and cooperation of the Internet of Content, Internet of Services, Internet of Things, and 3D interactive Internet, supported by an expanding network infrastructure foundation [6]. In ARLES, we chose to pay special attention to the Internet of Things (IoT). IoT is characterized by the integration of large numbers of real-world objects (or “things") onto the Internet, with the aim of turning high-level interactions with the physical world into a matter as simple as is interacting with the virtual world today. As such, two devices that will play a key role in the IoT are *sensors* and *actuators*. In fact, such devices are already seeing widespread adoption in the highly localized systems within our cars, mobile phones, laptops, home appliances, etc. In their current incarnation, however, sensors and actuators are used for little more than low-level inferences and basic services. This is partly due to their highly specialized domains (signal processing, estimation theory, robotics, etc.), which demand application programmers to also be domain experts, and partly due to a glaring lack of interconnectivity between all the different devices. Our work within this domain has been focused on two related directions:

- **Architecture of a Service Oriented Middleware for the Mobile Internet of Things**: Adopting the service-oriented architecture (SOA) approach towards middleware (see § 3.3), is an adequate solution towards addressing the heterogeneity and the unknown network topology issues in the IoT. SOA is commonly used in IoT solutions to abstract *things* or their measurements as services. The service-oriented paradigm decouples the functionalities of things from their hardware information or other technical details, and supports three core functionalities: *Discovery* and *Composition* of, and *Access* to services. Typically, in traditional uses of SOA, even if millions of services are registered, there is no need to select and access them all simultaneously. However, in the IoT, discovery, composition and access are undoubtedly more complicated. In fact, it is unlikely for a single or even a few services to be sufficient when providing real world measurements. In most cases, to accurately represent a real-world feature, a large number of services are selected to provide their measurements, and subsequently, all acquired values should be properly aggregated. As a consequence, discovery will return a large set of accessible services, redundant as they may be. Consumers are then expected to access the numerous providers to acquire their measurements, over which they should know the exact aggregation/fusion logic to apply. Furthermore, such logic requires precise knowledge and understanding of the real world and its governing physics and mathematics laws. Clearly, performing discovery, composition and access tasks as presented above incurs high communication and computation costs and is thus not realistic within the large scale IoT. In light of the above issues, we have been *revisiting the SOA and its interaction patterns* to support better scalability and exempt consumers from directly interacting with providers. Specifically, we introduce a **thing-based SOA** to wrap access and computation activities in a middleware that, unlike traditional SOA middleware, is aware of the real world, its physics and its mathematics rules; this has further led to our initial work on the components of such a middleware.
- **Probabilistic Registration for Large Scale Mobile Participatory Sensing**: An increasingly important component of the Internet of Things are modern smart phones, whose constituent sensors

and wireless connectivity make them ideal candidates for *mobile participatory sensing*, which aids in providing increased knowledge about the real world while relying on a large number of mobile devices. Those devices can host different types of sensors incorporated in every aspect of our lives. However, given the increasing number of capable mobile devices, any participatory sensing approach should be, first and foremost, *scalable*. To address this challenge, we present an approach to decrease the participation of (sensing) devices in a manner that does not compromise the accuracy of the real-world information while increasing the efficiency of the overall system. To reduce the number of the devices involved, we present a probabilistic registration approach [20], based on a realistic human mobility model, that allows devices to decide whether or not to register their sensing services depending on the probability of other, equivalent devices being present at the locations of their expected path. We used our techniques as the basis of the design and implementation of a registration middleware, using which mobile devices can base their registration decision. Through experiments performed on real and simulated datasets, we show that our approach scales, while not sacrificing significant amounts of sensing coverage.

Our IoT middleware is currently being used by the industrial partners in the FP7 IP CHOReOS project. Complementary to our research on this service oriented middleware for the Internet of Things, we have also been working on suitable abstractions for enabling easy application development for the IoT, discussed next.

6.5. Composing Applications in the Internet of Things

Participants: Peter Sawyer, Pankesh Patel, Animesh Pathak.

As introduced above, the Internet of Things integrates the physical world with the existing Internet, and is rapidly gaining popularity, thanks to the increased adoption of smart phones and sensing devices. Several IoT applications have been reported in recent research, and we expect to see increased adoption of IoT concepts in the fields of personal health, inventory management, and domestic energy usage monitoring, among others.

An important challenge to be addressed in the domain of IoT is to enable domain experts (health-care professionals, architects, city planners, etc.) to develop applications in their fields rapidly, with minimal support from skilled computer science professionals. Similar challenges have already been addressed in the closely related fields of Wireless Sensor and Actuator Networks (WSANs) and Pervasive/Ubiquitous computing. While the main challenge in the former is the *large scale* of the systems (hundreds to thousands of largely similar nodes, sensing and acting on the environment), the primary concern in the latter has been the *heterogeneity* of nodes and the major role that the user's own interaction with these nodes plays in these systems (cf. the classic "smart home" scenario where the user interacts with a smart display which works together with his refrigerator and toaster). The upcoming field of IoT includes both WSANs as well as smart appliances, in addition to the elements of the "traditional" Internet such as Web and database servers, exposing their functionalities as Web services, etc. Consequently, an ideal application development abstraction of the IoT will allow (domain expert) developers to intuitively specify the rich interactions between the extremely large number of disparate devices in the future Internet of Things [19].

The larger goal of our research is to propose a suitable application development framework which addresses the challenges introduced above. To that end, our work this year covered the following related areas:

- **Multi-stage Model-driven approach for IoT Application Development**: We have proposed a multi-stage model-driven approach for IoT application development based on a precise definition of the role to be played by each stakeholder involved in the process – domain expert, application designer, application developer, device developer, and network manager. The metamodels/abstractions available to each stakeholder are further customized using the inputs provided in the earlier stages by other stakeholders. We have also implemented code-generation and task-mapping techniques to support our approach. Our initial evaluation based on two realistic scenarios shows that the use of our techniques/framework succeeds in improving productivity in the IoT application development process.

- **Revisiting Requirements Engineering (RE) Practices for IoT**: Requirements engineering (RE) has evolved to discover, model, specify and manage the required and desired properties of software systems. Conventional RE makes an assumption that the knowledge from which the requirements will be formulated exists a-priori, even though the knowledge may be fragmentary, distributed and tacit. Thus, although their discovery may take significant effort, the requirements are discoverable using the appropriate RE practices.

  However, the last decade or so has seen the emergence of new types of systems where this assumption does not hold, including the IoT. Conventional RE is ill-equipped to discover, model, specify and manage these systems' requirements because incomplete knowledge of the context under which they must operate is available at design time. While some progress has been made, by (e.g.) maintaining requirements models that support reasoning over context at runtime, the IoT has now emerged to compound the challenge for RE. Drawing on experiences from ubiquitous computing and WSAN domains, in [22] we provided initial insights into how the field of RE needs to evolve in order to address the challenges brought forth by IoT.

We have incorporated our continued research in the above areas into *Srijan* (§ 5.5), which provides an easy-to-use graphical front-end to the various steps involved in developing an application using the ATaG macroprogramming framework.

6.6. Requirements-aware Systems for Self-adaptation under Uncertainty

Participants: Romina Torres, Nelly Bencomo, Valérie Issarny, Peter Sawyer.

The development of software-intensive systems is driven by their requirements. Traditional requirements engineering (RE) methods focus on resolving ambiguities in requirements and advocate specifying requirements in sufficient detail so that the implementation can be checked against them for conformance. In an ideal world, this way of thinking can be very effective. Requirements can be specified clearly, updated as necessary, and evolutions of the software design can be made with the requirements in mind.

Increasingly, however, it is not sufficient to fix requirements statically because they will change at runtime as the operating environment changes. Furthermore, as software systems become more pervasive, there is growing uncertainty about the environment and so requirements changes cannot be predicted at design-time. It is considerations such as these that have led to the development of self-adaptive systems (SASs), which have the ability to dynamically and autonomously reconfigure their behavior to respond to changing external conditions.

The key argument of our research is that current software engineering (SE) methods do not support well the kind of dynamic appraisal of requirements needed by a SAS. definition and structure of requirements is lost as requirements are refined into an implementation. Even in cases where requirements monitoring is explicitly included, high-level system requirements must be manually refined into low-level runtime artefacts during the design process so that they can be monitored. There is a lack of approaches supporting for runtime representation, evolution and assessment of requirements. Currently, the approaches mainly assume that it is possible to predefine and envisage the requirements for the total set of target behaviours. Such estimations and beliefs may not be appropriate, if the system is to recover during execution from unforeseen situations, or adapt dynamically to new environmental conditions or to satisfy new requirements that were not foreseen during development. A self-adaptive system is able, at run time, to satisfy new requirements and behaviors. Our research focuses on approaches to support the runtime representation of requirements that will underpin the way a system can reason and assess them during execution.

Our research has been carried out within the research project Marie Curie Fellowship called Requirements-aware Systems (nickname: Requirements@run.time). The research is based on a new paradigm for SE, called requirements-awareness (also known as requirements reflection), in which requirements are reified as runtime entities. Requirements-awareness allows systems to dynamically reason about themselves at the level of the requirements - in much the same way that architectural reflection currently allows runtime reasoning at the level of software. We believe that requirements-awareness (i.e. requirements reflection) will support the

development and management of SASs because it will raise the level of discourse at which a software system is able to reflect upon itself.

In the above context, we have been working on the design and implementation of systems with the ability to dynamically observe and reason about their requirements. The results will contribute towards the development of conceptual foundations, engineering techniques, and computing infrastructure for the access and manipulations of runtime abstractions of requirements. Currently, a prototype for the use of runtime goals has been developed. The RELAX language has been proposed to make requirements more tolerant to environmental uncertainty. Design assumptions, called Claims), are applied as markers of uncertainty that document how design assumptions affect goals. Monitoring Claims at runtime has been used to drive self-adaptation. By monitoring Claims during the execution of the systems, their veracity can be tested. If a Claim is falsified, the effect can be propagated to the system's goal model and an alternative (more suitable) means of goal realization will be selected, resulting in dynamic adaptation of the system to a configuration that better satisfies the goals under the prevailing environmental context.

7. Partnerships and Cooperations

7.1. National Grants

7.1.1. ANR

7.1.1.1. ANR MURPHY: Dependability-focused Evaluation of Sensor Networks

Participant: Animesh Pathak [correspondent].

- **Name:** MURPHY – *Dependability-focused Evaluation of Sensor Networks*
- **Related activities:** § 6.5
- **Period:** [January 2011 – December 2013]
- **Partners:** CNAM, Inria ARLES, LAAS - CNRS, SmartGrains, Univ. Valenciennes.

Murphy aims at easing the development of dependable and pervasive applications built on top of robust wireless sensor networks, thus providing a mean for early detection of possible failures, by estimating dependability metrics. This endeavor is undertaken by providing:

- Fault detection based on in-network event processing,
- Fault injection which attempts to accelerate the occurrence of faults so as to judge the quality of the error handling and hence, facilitate the evaluation of dependability,
- Advanced code dissemination across sensor networks, which is intended to (i) enable the dynamic and distributed insertion of faults and (ii) hide from the end user the complexity related to this task,
- Suitable abstractions to reason on faults, wireless sensor networks, data-centric and event-driven applications.

The aforementioned components enable to detect faults, diagnose possible causes and select appropriate corrective actions, and therefore to consolidate the dependability of sensor applications.

7.1.2. Inria Support

7.1.2.1. Inria D2T Action de Developpement Technologique MobiTools

Participants: Valérie Issarny, Bachir Moussa Tari Bako.

- **Name:** *MobiTools – Environnement de développement logiciel pour plateforme mobiles*
- **Period:** [January 2011 – December 2012]
- **Partners:** Inria (CRI Paris-Rocquencourt, EPI ARLES)

As part of the development of our software prototypes, MobiTools focuses on setting a supporting continuous integration platform (compilation, test, profiling, quality).

7.1.2.2. Inria D2T Action de Developpement Technologique Yarta

Participants: Animesh Pathak, George Rosca.

- **Name:** *Yarta – Middleware for mobile social ecosystems*
- **Period:** [October 2012 – September 2013]
- **Partners:** Inria (CRI Paris-Rocquencourt, EPI ARLES)

This project targets the development of Yarta, a middleware for managing mobile social ecosystems, which builds upon existing research in context-awareness in the pervasive computing domain. The work involves development effort in the multi-layer middleware architecture of Yarta, providing the needed functionalities, including *i*) Storage of social data in an interoperable format, using semantic technologies such as RDF; *ii*) Extraction of social ties from context (both physical and virtual); *iii*) Enforcement of access control to protect social data from arbitrary access; and *iv*) A rich set of mobile social ecosystem (MSE) management functionalities, using which mobile social applications can be developed. Specifically, the ADT will be used to support the public open source release and evolution of the Yarta middleware, which is currently a research prototype.

7.2. European Initiatives

7.2.1. FP7 Projects

7.2.1.1. FP7 ICT FET IP CONNECT

Participant: Valérie Issarny [correspondent].

Name: CONNECT – *Emergent Connectors for Eternal Software Intensive Networked Systems*

URL: http://www.connect-forever.eu/

Type: COOPERATION (ICT)

Defi: ICT forever yours

Instrument: Integrated Project (IP)

Related activities: § 6.2

Period: [February 2009 - November 2012]

Partners: Inria (CRI Paris-Rocquencourt) [**project coordinator**], Ambientic (France), CNR (Italy), DoCoMo (Germany), Lancaster University (UK), Thales Communications SA (France), Universita degli Studi L'Aquila (Italy), Technische Universitaet Dortmund (Germany), University of Oxford (UK), Uppsala Universitet (Sweden), Peking University (China).

The CONNECT Integrated Project aims at enabling continuous composition of networked systems to respond to the evolution of functionalities provided to and required from the networked environment. At present the efficacy of integrating and composing networked systems depends on the level of interoperability of the systems's underlying technologies. However, interoperable middleware cannot cover the ever growing heterogeneity dimensions of the networked environment. CONNECT aims at dropping the interoperability barrier by adopting a revolutionary approach to the seamless networking of digital systems, that is, synthesizing on the fly the connectors via which networked systems communicate. The resulting emergent connectors are effectively synthesized according to the behavioral semantics of application- down to middleware-layer protocols run by the interacting parties. The synthesis process is based on a formal foundation for connectors, which allows learning, reasoning about and adapting the interaction behavior of networked systems at run-time. Synthesized connectors are concrete emergent system entities that are dependable, unobtrusive, and evolvable, while not compromising the quality of software applications. To reach these objectives the CONNECT project undertakes interdisciplinary research in the areas of behavior learning, formal methods, semantic services, software engineering, dependability, and middleware. Specifically, CONNECT investigates the following issues and related challenges: (i) Modeling and reasoning about peer system functionalities, (ii) Modeling and reasoning about connector behaviors, (iii) Runtime synthesis of connectors, (iv) Learning connector behaviors, (v) Dependability assurance, and (vi) System architecture. The effectiveness of CONNECT research is assessed by experimenting in the field of wide area, highly heterogeneous systems where today's solutions to interoperability already fall short (e.g., systems of systems).

7.2.1.2. FP7 ICT IP CHOReOS

Participants: Nikolaos Georgantas [correspondent], Valérie Issarny [correspondent].

Name: CHOReOS – *Large Scale Choreographies for the Future Internet*

URL: http://www.choreos.eu/

Type: COOPERATION (ICT)

Defi: Internet of Services, Software & Virtualisation

Instrument: Integrated Project (IP)

Related activities: § 6.3 & § 6.4

Period: [February October 2010 - September 2013]

Partners: NoMagic Europe (Lithuania), CEFRIEL (Italy), CNR (Italy), Linagora (France), Inria (CRI Paris-Rocquencourt) [**scientific leader**], MLS Multimedia A.E. (Greece), OW2 Consortium, Thales Communications S.A. (France) [**coordinator**], The City University, London (UK), Università degli Studi dell'Aquila (Italy), Universidade de São Paulo (Brazil), University of Ioannina (Greece), SSII VIA (Latvia), Virtual Trip Ltd. (Greece), Wind Telecommunicazioni S.p.A (Italy).

CHOReOS aims at assisting the engineering of software service compositions in the revolutionary networking environment created by the Future Internet. Indeed, sustaining service composition and moving it closer to the end users in the Future Internet is a prime requirement to ensure that the wealth of networked services will get appropriately leveraged and reused. This again stresses the required move from static to dynamic development, effectively calling for adequate support for service reuse; much like software reuse has been a central concern in software engineering over the last two decades. This is why CHOReOS adopts the Service Oriented Computing (SOC) paradigm, where networked resources are abstracted as services so as to ease their discovery, access and composition, and thus reuse. However, although latest advances in the SOC domain enable facing (at least partly) the requirements of today's Internet and related networking capabilities, engineering service compositions in the light of the Future Internet challenges — in particular the ultra large scale (ULS) on all imaginable dimensions as well as the evolution of the development process from a mostly static process to a dynamic user-centric one — is far from adequately addressed. Therefore, the CHOReOS goal is to address these challenges by devising a dynamic development process, and associated methods, tools and middleware, to sustain the composition of services in the Future Internet.

7.2.1.3. FP7 PEOPLE Requirements@run.time

Participant: Nelly Bencomo [correspondent].

Name: Requirements@run.time: *Requirements-aware systems*

URL: https://www-roc.inria.fr/arles/index.php/members/220-marie-curie-project-requirements-aware-systems-requirementsruntime

Type: PEOPLE

Instrument: Marie Curie Intra-European Fellowships for Career Development (IEF)

Related activities: § 6.6

Period: [May 2011 - May 2013]

Partners: Inria (CRI Paris-Rocquencourt).

This project uses the novel notion of requirements reflection, that is, the ability of a system to dynamically observe and reason about its requirements. It aims to address the need of having systems requirements-aware by reifying requirements as run-time objects (i.e. requirements@run.time). These systems provide a runtime model of their requirements that allow them to reason, evaluate and report on their conformance to their requirements during execution.This project contributes towards development of conceptual foundations, engineering techniques, and computing infrastructure for the systematic development of dynamically-adaptive systems based on the principle of requirements reflection The researchers build upon their extensive expertise in the area of reflective middleware and reflective architectures and research projects like CONNECT.

7.2.1.4. FP7 ICT NoE NESSoS

Participants: Valérie Issarny [correspondent], Animesh Pathak [correspondent].

Name: NESSoS – *Network of Excellence on Engineering Secure Future Internet Software Services and Systems*

URL: http://www.nessos-project.eu

Type: COOPERATION (ICT)

Defi: Trustworthy ICT

Instrument: Network of Excellence (NoE)

Related activities: § 6

Period: [October 2010 - March 2013]

Partners: Atos Origin (Spain), CNR (Italy) [**coordinators**], ETH Zürich (Switzerland), IMDEA Software (Spain), Inria (EPI ARLES, CASSIS, and TRISKELL), KU Leuven (Belgium), LMU München (Germany), Siemens AG (Germany), SINTEF (Norway), University Duisburg-Essen (Germany), Universidad de Malaga (Spain), Università degli studi di Trento (Italy).

The Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS) aims at constituting and integrating a long lasting research community on engineering secure software-based services and systems. The NESSoS engineering of secure software services is based on the principle of addressing security concerns from the very beginning in system analysis and design, thus contributing to reduce the amount of system and service vulnerabilities and enabling the systematic treatment of security needs through the engineering process. In light of the unique security requirements exposed by the Future Internet, new results are achieved by means of an integrated research, as to improve the necessary assurance level and to address risk and cost during the software development cycle in order to prioritize and manage investments. NESSoS integrates the research labs involved; NESSoS re-addresses, integrates, harmonizes and fosters the research activities in the necessary areas, and increases and spreads the research excellence. NESSoS also impacts training and education activities in Europe to grow a new generation of skilled researchers and practitioners in the area. NESSoS collaborates with industrial stakeholders to improve the industry best practices and support a rapid growth of software-based service systems in the Future Internet.

7.2.1.5. FP7 ICT CA EternalS

Participant: Valérie Issarny [correspondent].

Name: EternalS – *Trustworthy Eternal Systems via Evolving Software, Data and Knowledge*

URL: http://www.eternals.eu

Type: CAPACITIES (ICT)

Defi: FET - Proactive

Instrument: Coordination and Support Action (CSA)

Related activities: § 6.2

Period: [March 2010 - February 2013]

Partners: Inria (CRI Paris-Rocquencourt), KU Leuven (Belgium), Queen Mary University (UK), University of Chalmers (Sweden), University of Trento (Italy), Waterford Institute of Technology (Ireland).

Latest research work within ICT has allowed to pinpoint the most important and urgently required features that future systems should possess to meet users' needs. Accordingly, methods making systems capable of adapting to changes in user requirements and application domains have been pointed out as key research areas. Adaptation and evolution depend on several dimensions, e.g., time, location, and security conditions, expressing the diversity of the context in which systems operate. A design based on an effective management of these dimensions constitutes a remarkable step toward the realization of Trustworthy Eternal Systems. The EternalS Coordination Action specifically aims at coordinating research in that area based on a researcher Task Force together with community building activities, where the organization of large workshops and conferences is just one of the tools that will be used to conduct a successful CA.

7.3. International Initiatives

7.3.1. Participation In International Programs

7.3.1.1. Project M@TURE – International scientific cooperation programme Inria/Brazil

Participant: Nikolaos Georgantas [Correspondant].

Name: M@TURE – Models @ runtime for self-adaptive pervasive systems: enabling user-in-the-loop, requirement-awareness, and interoperability in ad hoc settings

Instrument: Inria-Brazil cooperation programme

Period: [October 2012 – September 2014]

Partners: Joint project with Institute of Informatics, Federal University of Goias, Brazil.

The overall goal of the M@TURE project is to design, implement and evaluate a novel approach and architecture – comprising conceptual foundations, engineering techniques, and supporting middleware infrastructure – for self-adaptive pervasive systems by building on the notion of Models@run.time. Models@run.time extends the applicability of models and abstractions to the runtime environment. In contrast to design-time models, runtime models are used to reason about the running system taking into account its operating environment, and thus these models enable automating runtime decisions and actions regarding the creation, configuration, and evolution of the system. We will in particular focus on the following dimensions and related models: (i) Requirements models making a system requirements-aware at runtime; (ii) Application- and middleware-level interoperability models exposing to an external observer the technological and business features of a system; and (iii) End-user and system engineer models modeling the internal elements of a system at two different abstraction levels. These models will be considered both independently and, more importantly, in synergy in order to introduce a comprehensive conceptual and architectural solution for self-adaptive pervasive systems.

7.4. International Research Visitors

7.4.1. Visits of International Scientists

7.4.1.1. Internships

Amel Belaggoun (from Apr 2012 until Sep 2012)

Subject: Exploring the Use of Dynamic Decision Networks for Self-Adaptive Systems

Institution: Université de Versailles Saint-Quentin-en-Yvelines (France)

Ajay Chhatwal (from Jan 2012 until Mar 2012)

Subject: Supporting Application Development for the Future Internet of Smart Things and Services

Institution: Indian Institute of Technology, Banaras Hindu University, Varanasi (India)

Guilherme Nogueira (from Apr 2012 until Oct 2012)

Subject: Facilitating the Specification of Fault Tolerance Requirements in Sensor Network Macroprograms

Institution: University of São Paulo (Brazil)

8. Dissemination

8.1. Scientific Animation

8.1.1. Programme Committees of Conferences and Workshops

- Nelly Bencomo is PC member of the following international conferences: SEAMS '12& '13, MODELS '12& '13, CIbSE '12& '13, Poster track at RE '12& '13, SANES session at PECCS '13, SBES '13;
- Nelly Bencomo is PC member of the following international workshops: UsARE '12 at ICSE '12, VAMOS '13;
- Nikolaos Georgantas is PC member of the following international conferences: DAIS '12, ICSOFT '12, DATA '12, SOSE '13, ANT '13;
- Nikolaos Georgantas is PC member of the following international workshops: MW4NG'12, ARM'12;
- Valérie Issarny is PC member of the following international conferences: ESSOS '12, FASE '13, ICDCS '13, ICSE-NIER '12, ICSE '13, IFIPTM '13, ISARCS '13, Middleware '12, SEAMS '12& '13;
- Valérie Issarny is PC member of the following international workshops: SESO@ICSE '13, SERENE' 12& '13;
- Animesh Pathak is PC member of the following international conferences: ICCC '12, iThings '12, NPC '12, SENSORNETS '12, S-Cube '12; and
- Animesh Pathak is PC member of the following international workshops: SESENA '12, WSN4ITS '12, GeoHCI '13, and the Middleware '12 doctoral symposium.

8.1.2. Leadership Services in Academic Events and Edited Journals

- Nelly Bencomo is Student Volunteers Co-Chair at ICSE 2012;
- Nikolaos Georgantas is associate editor of the International Journal of Ambient Computing and Intelligence (IJACI);
- Nikolaos Georgantas is associate editor of the international journal Conference Papers in Computer Science;
- Valérie Issarny is member of the Steering Committee of the Middleware and ESEC/FSE conferences;
- Valérie Issarny is associate editor of the Springer JISA Journal of Internet Services and Applications;
- Valérie Issarny is the general chair of the ESSoS 2013 International Symposium on Engineering Secure Software and Systems; and
- Animesh Pathak is the Production Chair of HiPC 2012 & 2013.

8.1.3. Other Academic Services

- Nikolaos Georgantas is member of the PhD monitoring committee at Inria Paris-Rocquencourt;
- Valérie Issarny acted as Paris Node director of the EIT ICT Labs KIC from January to September 2012;
- Valérie Issarny is coordinator of the EC FP7 FET IP CONNECT;
- Valérie Issarny is scientific leader of the EC FP7 IP CHOReOS;
- Valérie Issarny is expert for the EC FP7 ICT work programme, and ARTEMIS Joint Undertaking;
- Valérie Issarny is member of the INRETS scientific council (IFSTTAR since 2011) & "Commission d'évaluation des chercheurs";
- Valérie Issary is member of the GDR GPL scientific council;
- Valérie Issarny is member of the evaluation committee of ANR BlancJCJC'12, of Swedish Research Council subcommittee for Computer Science'2012 and Vienna Science and Technology Fund'2012; and
- Animesh Pathak is member of review board for EIT ICT labs 2013 call, Smart Spaces, and Cyber-Physical Systems tracks.

8.2. Teaching - Supervision - Juries

8.2.1. Teaching

Valérie Issarny gives a 12 hours lecture on “Pervasive Service Oriented Computing" as part of the SWAS lecture of the Master 2 COSY of the University of Versailles Saint-Quentin en Yvelines;

Animesh Pathak co-taught a M2 level course on “Web sémantique, contenus et usages” at University of Versailles Saint-Quentin-en-Yvelines in Spring 2012; and

Animesh Pathak gave two three-hour guest lectures at CNAM, Paris as part of a M2 level course.

8.2.2. Supervision

In 2012, the following student successfully defended his PhD: Nebil Ben Mabrouk, *“QoS Service Oriented Middleware for Pervasive environments”*, defended April 2012, advised by Nikolaos Georgantas and Valérie Issarny

Additionally, the following PhD theses are currently in progress at the ARLES project-team:

- Emil Andriescu, *“Synthèse et déploiement dynamiques de protocoles de médiation pour l’interopérabilité dans les environnements collaboratifs nomades”*, started October 2012, advised by Valérie Issarny
- Dionysis Athanasopoulos, *“Evolution and Maintenance in Service-oriented Software”*, started October 2008, advised by Apostolos Zarras and Valérie Issarny
- Amel Bennaceur, *“Synthèse dynamique de connecteurs pour les réseaux ubiquitaires"*, started October 2009, advised by Valérie Issarny
- Benjamin Billet, *“Self-adaptive Streaming Middleware for the Internet of Things”*, started October 2012, advised by Valérie Issarny
- Sara Hachem, *“Middleware pour l’Internet des objets intelligents"*, started October 2012, advised by Animesh Pathak and Valérie Issarny
- Pankesh Patel, *“Enabling High-Level Application Development in Sensing-and-Actuation-augmented Pervasive Systems*, started October 2010, advised by Animesh Pathak and Valérie Issarny

8.2.3. Juries

Valérie Issarny served on the PhD jury of two students: Ahmed Gater (May 2012, UVSQ), and Ajay Kattepur (November 2012, U. Rennes I)

9. Bibliography

Major publications by the team in recent years

[1] S. BEN MOKHTAR, D. PREUVENEERS, N. GEORGANTAS, V. ISSARNY, Y. BERBERS. *EASY: Efficient SemAntic Service DiscoverY in Pervasive Computing Environments with QoS and Context Support*, in "Journal of Systems and Software, Special Issue on Web Services Modelling and Testing", 2008, vol. 81, n^{o} 5, p. 785-808.

[2] G. BLAIR, A. BENNACEUR, N. GEORGANTAS, P. GRACE, V. ISSARNY, V. NUNDLOLL, M. PAOLUCCI. *The Role of Ontologies in Emergent Middleware: Supporting Interoperability in Complex Distributed Systems*, in "Big Ideas track of ACM/IFIP/USENIX 12th International Middleware Conference", Lisbon, Portugal, 2011, http://hal.inria.fr/inria-00629059/en.

[3] M. CAPORUSCIO, P.-G. RAVERDY, V. ISSARNY. *ubiSOAP: A Service Oriented Middleware for Ubiquitous Networking*, in "IEEE Transactions on Services Computing", 2010, vol. 99, n° PrePrints [*DOI :* 10.1109/TSC.2010.60], http://hal.inria.fr/inria-00519577.

[4] D. CHARLET, V. ISSARNY, R. CHIBOUT. *Energy-efficient middleware-layer multi-radio networking: An assessment in the area of service discovery*, in "Comput. Netw.", January 2008, vol. 52, p. 4–24 [*DOI :* 10.1016/J.COMNET.2007.09.018], http://dl.acm.org/citation.cfm?id=1321781.1321914.

[5] V. ISSARNY, M. CAPORUSCIO, N. GEORGANTAS. *A Perspective on the Future of Middleware-based Software Engineering*, in "FOSE '07: 2007 Future of Software Engineering", Washington, DC, USA, IEEE Computer Society, 2007, p. 244–258, http://dx.doi.org/10.1109/FOSE.2007.2.

[6] V. ISSARNY, N. GEORGANTAS, S. HACHEM, A. ZARRAS, P. VASSILIADIS, M. AUTILI, M. A. GEROSA, A. BEN HAMIDA. *Service-Oriented Middleware for the Future Internet: State of the Art and Research Directions*, in "Journal of Internet Services and Applications", May 2011, vol. 2, n° 1, p. 23-45 [*DOI :* 10.1007/s13174-011-0021-3], http://hal.inria.fr/inria-00588753/en.

Publications of the year

Doctoral Dissertations and Habilitation Theses

[7] N. BEN MABROUK. *QoS-aware Service-Oriented Middleware for Pervasive Environments*, Université Pierre et Marie Curie - Paris VI, April 2012.

Articles in International Peer-Reviewed Journals

[8] N. BENCOMO, A. BENNACEUR, P. GRACE, G. BLAIR, V. ISSARNY. *The Role of Models@run.time in Supporting On-the-fly Interoperability*, in "Computing", October 2012, http://hal.inria.fr/hal-00733338.

[9] N. BENCOMO, S. HALLSTEINSEN, E. ALMEIDA. *A View of the Landscape of Dynamic Software Product Lines*, in "IEEE Computing, Special Issue on Dynamic Software Product Lines", October 2012, http://hal.inria.fr/hal-00722377.

[10] M. CAPORUSCIO, P.-G. RAVERDY, V. ISSARNY. *ubiSOAP: A Service Oriented Middleware for Ubiquitous Networking*, in "IEEE Transactions on Services Computing", 2012.

[11] P. SAWYER, R. MAZO, D. DIAZ, C. SALINESI, D. HUGHES. *Using Constraint Programming to Manage Configurations in Self-Adaptive Systems*, in "Computer", 2012, vol. 45, n° 10.

[12] I. SCHAEFER, R. RABISER, D. CLARKE, L. BETTINI, D. BENAVIDES, G. BOTTERWECK, A. PATHAK, S. TRUJILLO, K. VILLELA. *Software diversity: state of the art and perspectives*, in "International Journal on Software Tools for Technology Transfer", October 2012, vol. 14, n° 5, p. 477-495 [*DOI :* 10.1007/s10009-012-0253-y], http://hal.inria.fr/hal-00723754.

International Conferences with Proceedings

[13] D. ATHANASOPOULOS, A. ZARRAS, P. VASSILIADIS. *Service Selection for Happy Users: Making User Intuitive Quality Abstractions*, in "SIGSOFT/FSE'12 20th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-20)", North Carolina, United States, W. TRACZ (editor), ACM, November 2012, http://hal.inria.fr/hal-00728844.

[14] N. BENCOMO, A. BELAGGOUN. *Supporting Decision-making for Self-adaptive Systems: From Goal Models to Dynamic Decision Networks*, in "9th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ)", 2013.

[15] N. BENCOMO, K. WELSH, P. SAWYER, J. WHITTLE. *Towards Self-explanation in Adaptive Systems Assumptions*, in "17th IEEE International Conference International Conference on Engineering of Complex Computer Systems ICECCS 2012", Paris, France, July 2012, http://hal.inria.fr/hal-00719001.

[16] A. BENNACEUR, V. ISSARNY, R. SPALAZZESE, S. TYAGI. *Achieving Interoperability through Semantics-based Technologies: The Instant Messaging Case*, in "ISWC 2012 - 11th International Semantic Web Conference", Boston, United States, 2012, http://hal.inria.fr/hal-00721280.

[17] A. BENNACEUR, V. ISSARNY, D. SYKES, F. HOWAR, M. ISBERNER, B. STEFFEN, J. RICHARD, M. ALESSANDRO. *Machine Learning for Emergent Middleware*, in "JIMSE- Joint workshops on Intelligent Methods for Software System Engineering", Montpellier, France, 2012, http://hal.inria.fr/hal-00722051.

[18] L. CAVALLARO, P. SAWYER, D. SYKES, N. BENCOMO, V. ISSARNY. *Satisfying Requirements for Pervasive Service Compositions*, in "7th International Workshop on Models@run.time (MRT 2012)", Innsbruck, Austria, ACM, October 2012, http://hal.inria.fr/hal-00733346.

[19] A. CHHATWAL, K. K. SHUKLA, A. PATHAK. *Application Development for the Internet of Things: Observations and Challenges*, in "Work in Progress session of IEEE International Conference on the Internet of Things, iThings 2012", Besançon, France, 2012, http://hal.inria.fr/hal-00763121.

[20] S. HACHEM, A. PATHAK, V. ISSARNY. *Probabilistic Registration for Large-Scale Mobile Participatory Sensing*, in "IEEE International Conference on Pervasive Computing", California, USA, March 2013, http://hal.inria.fr/hal-00769087.

[21] A. RAMIREZ, B. CHENG, N. BENCOMO, P. SAWYER. *RELAXing Claims: Coping With Uncertainty While Evaluating Assumptions at Run Time*, in "ACM/IEEE 15th International Conference on Model Driven Engineering Languages & Systems MODELS 2012 MODELS 2012", InnsBruck, Austria, September 2012, http://hal.inria.fr/hal-00718997.

[22] P. SAWYER, A. PATHAK, N. BENCOMO, V. ISSARNY. *How the Web of Things Challenges Requirements Engineering*, in "3rd Workshop on The Web and Requirements Engineering at 12th International Conference on Web Engineering ICWE 2012", Berlin, Germany, July 2012, http://hal.inria.fr/hal-00722382.

[23] R. TORRES, N. BENCOMO, H. ASTUDILLO. *Mitigating the obsolescence of quality specifications models in service-based systems*, in "Workshop MoDRE 2012, Model-driven Requirements Engineering Workshop", Chicago, United States, September 2012, http://hal.inria.fr/hal-00719049.

Scientific Books (or Scientific Book chapters)

[24] J. ANDERSSON, L. BARESI, N. BENCOMO, R. DE LEMOS, A. GORLA, P. INVERARDI, T. VOGEL. *Software Engineering Processes for Self-adaptive Systems*, in "Software Engineering for Self-adaptive Systems 2", R. DE LEMOS, H. GIESE, H. MULLER, M. SHAW (editors), Lecture Notes in Computer Science, Springer, September 2012, vol. 7475, http://hal.inria.fr/hal-00719003.

[25] R. DE LEMOS, H. GIESE, H. MÜLLER, M. SHAW, J. ANDERSSON, L. BARESI, B. BECKER, N. BENCOMO, Y. BRUN, B. CIKIC, R. DESMARAIS, S. DUSTDAR, G. ENGELS, K. GEIHS, K. GOESCHKA, A. GORLA, V. GRASSI, P. INVERARDI, G. KARSAI, J. KRAMER, M. LITOIU, A. LOPES, J. MAGEE, S. MALEK, S. MANKOVSKII, R. MIRANDOLA, J. MYLOPOULOS, O. NIERSTRASZ, M. PEZZE, C. PREHOFER, W. SCHAFER, W. SCHLICHTING, B. SCHMERL, D. SMITH, J. SOUSA, G. TAMURA, L. TAHVILDARI, N. VILLEGAS, T. VOGEL, D. WEYNS, K. WONG, J. WUTTKE. *Software Engineering for Self-Adaptive Systems: A Second Research Roadmap*, in "Software Engineering for Self-Adaptive Systems II", R. DE LEMOS, H. GIESE, H. MÜLLER, M. SHAW (editors), Springer, September 2012, http://hal.inria.fr/hal-00719006.

[26] A. B. HAMIDA, F. KON, G. A. OLIVA, C. E. M. D. SANTOS, J.-P. LORRÉ, M. AUTILI, G. DE ANGELIS, A. ZARRAS, N. GEORGANTAS, V. ISSARNY, A. BERTOLINO. *An Integrated Development and Runtime Environment for the Future Internet*, Future Internet Assembly, Springer, 2012, p. 81-92.

Books or Proceedings Editing

[27] V. ISSARNY, G. BLAIR (editors). *Journal of Internet Services and Applications, Special Issue on Future of Middleware (FOME)*, Springer, May 2012, vol. 3, n^{o} 1.

Scientific Popularization

[28] A. BENNACEUR, P. INVERARDI, V. ISSARNY, R. SPALAZZESE. *Automated Synthesis of CONNECTors to support Software Evolution*, in "ERCIM News", January 2012, http://hal.inria.fr/hal-00662058.

[29] P. GRACE, G. BLAIR, V. ISSARNY. *Emergent Middleware*, in "ERCIM News", January 2012, vol. 2012, n^{o} 88, p. 27-28, http://hal.inria.fr/hal-00659762.

Other Publications

[30] *CHOReOS consortium. CHOReOS Deliverables. IP CHOReOS EU project, FP7 grant agreement number 257178*, 2012, http://www.choreos.eu/.

[31] CONNECT *consortium.* CONNECT *Deliverables. FET IP* CONNECT *EU project, FP7 grant agreement number 231167*, 2012, http://www.connect-forever.eu/.