INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team ARLES

Software Architectures and Distributed Systems

Paris - Rocquencourt

Theme : Distributed Systems and Services

Activity Report

2009

Table of contents

1. Team

Research Scientist

Valérie Issarny [Team Leader, INRIA Research Director, HdR]
Nikolaos Georgantas [INRIA Research Scientist]
Animesh Pathak [INRIA Research Scientist, since September 2009]

Technical Staff

Mohammad Ashiqur Rahaman [*Ingénieur Expert*, INRIA ITEmIS, since September 2009]
Sandrine Beauche [*Ingénieur Expert*, INRIA SemEUsE]
Amel Bennaceur [*Ingénieur Jeune Diplomée*, INRIA, until September 2009]
Mauro Caporuscio [*Ingénieur Expert*, INRIA EXOTICUS, until January 2009]
Sneha Godbole [*Ingénieur Expert*, INRIA EXOTICUS, February 2009 – October 2009]
Sneha Godbole [*Ingénieur Expert*, INRIA SemEUsE, since November 2009]
Iraklis Leontiadis [*Ingénieur Jeune Diplomée*, INRIA, since October 2009]
Rachid Saadi [*Ingénieur Expert*, INRIA CONNECT, since September 2009]
Pushpendra Singh [*Ingénieur Expert*, INRIA EXOTICUS, until October 2009]
Roberto Speicys Cardoso [*Ingénieur Expert*, INRIA EXOTICUS, April 2009 – October 2009]
Roberto Speicys Cardoso [*Ingénieur Spécialiste*, INRIA, since November 2009]

PhD Student

Hamid Ameziani [INRIA ITEmIS, University Pierre et Marie Curie - Paris 6, since December 2009]
Dionysis Athanasopoulos [INRIA ForeverSOA, University of Ioannina, Greece, since October 2009]
Nebil Ben Mabrouk [INRIA CORDI, University Pierre and Marie Curie - Paris 6]
Amel Bennaceur [INRIA CONNECT, University Pierre et Marie Curie Paris - 6, since October 2009]
Manel Fredj [INRIA, University Pierre et Marie Curie - Paris 6, until January 2009]
Amir Seyedi [INRIA SemEUsE, University Pierre et Marie Curie - Paris 6, since December 2009]
Roberto Speicys Cardoso [INRIA, University Pierre et Marie Curie - Paris 6, until March 2009]

Post-Doctoral Fellow

Iyad Alshabani [INRIA, since January 2009]
Animesh Pathak [INRIA, until August 2009]
Alessandra Toninelli [ERCIM, since October 2009]

Visiting Scientist

Oleg Davidyuk [PhD Student, University of Oulu, Finland, until April 2009]
Romina Spalazzese [PhD Student, Universita' degli Studi dell'Aquila, Italy, October 2009 – November 2009]
Apostolos Zarras [Lecturer, University of Ioannina, Greece, June 2009 – July 2009]

Administrative Assistant

Sylvie Burini

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 communications technologies. The study of these systems forms the core of the ARLES' work, which is specifically concerned with defining new system software architectures, based on the use of emerging networking technologies.

In the recent years, middleware has emerged as a prominent solution to overcome the heterogeneity of the distributed infrastructure. It establishes a software layer that homogenizes the 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 to the development of distributed systems, which is increasingly complex, the role of middleware has proved central in the software system development practice.

However, the development of distributed systems using middleware remains a complex task[1]. In particular, middleware must define 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, new requirements arise for middleware, e.g., supporting open and mobile networking, and context awareness [6].

Additionally, the past years have seen a proliferation in middleware research, which had led to a problem of its own — the various middlewares are not compatible with each others, making *interoperability* an extremely important research area.

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. 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 extremtly 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. Our further goal is to assist the abovementioned engineering process.

2.2. Highlights of the Year

During this year, while we have been pursuing our research on advanced service-oriented architectures and related middleware solutions for next generation networking environments, we have been initiating research into two directions, both called for by the ongoing drastic evolution of the networking environment:

- High-level application development in heterogeneous sensor networks, focussing specifically on data-driven macroprogramming (§ 6.7), and
- 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 programme of the EC FP7-ICT (§ 6.5, § 7.1.1).

3. Scientific Foundations

3.1. Scientific Foundations

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 the environment. This leads us to investigate dedicated software architecture styles with a special emphasis on associated:

- *architectural models* for mobile distributed software systems, enabling pervasive computing, together with associated methods and tools for reasoning about the systems' behavior and automating the systems' composition at runtime, and
- *middleware platforms* for alleviating the complexity of systems development, by in particular offering adequate network abstractions.

The next section provides a brief overview of the state of the art in the area of software architectures for distributed systems; we survey base architectural styles that we consider in our work and further discuss the benefits of architecture-based development of distributed systems. Section 3.3 then addresses middleware architectures for mobile systems, discussing the impact of today's wireless networks on the software systems, and core requirements that we consider for the middleware, i.e., managing the network's dynamics and enforcing dependability for the mobile systems. Each section refers to results on which we build, and additionally discusses some of the research challenges that remain in the area and that we are investigating as part of our research.

3.2. Software Architectures for Distributed Systems

Architectural representations of systems have shown to be effective in assisting the understanding of broader system concerns by abstracting away from details of the system. This is achieved by employing architectural styles that are appropriate for describing systems in terms of *components*, the interactions between these components – *connectors* – and the properties that regulate the composition of components – *configurations*. Thus, components are units of computation or data store, while connectors are units of interaction among components or rules that govern the interactions. Defining notations for the description of software architectures has been one of the most active areas of research in the software architecture community since its emergence in the early 90's. Regarding the overall development process, *Architecture Description Languages* (ADLs) that have been proposed so far are mainly concerned with architecture modeling during the analysis and design phase. In addition, some existing ADLs enable the deriving of system implementation and deployment, provided there is an available implementation of the system's primitive components and connectors. In general, a major objective in the definition of ADLs is to provide associated Computer-aided Software Engineering (CASE) tools, which enables tasks underpinning the development process to be automated. In this context, special emphasis has been put on the usage of formal methods and associated tools for the analysis of complex software systems by focusing on the system's architecture, which is abstract and concise. As a result, work in the software architecture community provides a sound basis towards assisting the development of robust distributed systems, which is further eased by middleware platforms.

3.2.1. Middleware-based and Service-oriented Software Architectures

Available middleware can be classified into three main categories: transaction-oriented middleware that mainly aims at system architectures whose components are database applications; message-oriented middleware that targets system architectures whose component interactions rely on publish/subscribe communication schemes; and object-oriented middleware that is based on the remote procedure call paradigm and enables the development of system architectures complying with the object paradigm (*e.g.*, inheritance, state encapsulation), and, hence, enforces an object model for the system (i.e., the architectural components are objects). Development of middleware-based systems is now quite mature although middleware heterogeneity is still an open issue. In addition, dealing with middleware heterogeneity in the presence of dynamic composition raises the issue of dynamically integrating and possibly adapting the system's components, which is being investigated in the middleware community.

Evolution of middleware and distributed system technologies has further led to the emergence of service-oriented system architectures to cope with the requirements of Internet-based systems. *Software services*, in particular in the form of XML *Web services*, offer a promising paradigm for software integration and interoperation. Simply stated, a service is an instantiated configured system, which may be composed with other services to offer a new system that actually realizes a system of systems. Although the definition of the overall Web services architecture is still incomplete, the base standards defining a core middleware for Web services have already been released by the W3C[5], partly building upon results from object-based and component-based middleware technologies. These standards relate to the specification of Web services and supporting interaction protocols. Composing Web services relates to dealing with the assemblage of existing services, so as to deliver a new service, given the corresponding published interfaces. Integration of Web services is then realized according to the specification of the overall process composing the Web services. The process specifying the composition must not solely define the functional behavior of the process in terms of interactions with the composed services, but also the process' non-functional properties, possibly exploiting middleware-related services. Various non-functional properties (*e.g.*, availability, extensibility, reliability, openness, performance, security, scalability) should be accounted for in the context of Web services. However, enforcing dependability of composite Web services is one of the most challenging issues, especially for supporting business processes, due to the fact that the composition process deals with the assemblage of loosely-coupled autonomous components.

Although Web services have primarily been designed for realizing complex business processes over the Internet, they are a promising architectural choice for pervasive computing. The pervasiveness of the Web allows anticipating the availability of Web services in most environments, considering further that they may be hosted on mobile devices. Hence, this serves as a sound basis towards dealing with the dynamic composition of services in the pervasive computing environment. However, this further requires specification of the Web services' functional and non-functional behavior that can be exploited for their dynamic selection and integration, which may in particular build upon work on the Semantic Web.

3.2.2. Architecture-based Development of Distributed Systems

The building blocks of distributed software systems relying on some middleware platform, fit quite naturally with the ones of software architectures: the architectural components correspond to the application components managed by the middleware, and the architectural connectors correspond to the supporting middleware.

[5] http://www.w3.org

Hence, the development of such systems can be assisted with an architecture-based development process in a straightforward way. This approach is already supported by a number of ADL-based development environments targeting system construction, such as the Aster environment that was previously developed by members of the ARLES project-team.

However, most of the work on the specification of connectors has focused on the characterization of the interaction protocols among components, whilst connectors abstracting middleware embed additional complex functionalities (*e.g.*, support for provisioning fault tolerance, security, transactions). The above concern has led the software architecture community to examine the specification of the non-functional properties offered by connectors. For instance, these may be specified in terms of logic formulae, which further enables synthesizing middleware customized to the application requirements, as supported by the Aster ADL. Another issue that arises when integrating existing components, as provided by middleware platforms, results from assembling components that rely on distinct interaction patterns. This aspect is known as *architectural mismatch* and is one of the criteria substantiating the need for connectors as first-class entities in architecture descriptions. The abstract specification of connector behavior, as, for instance, supported by the Wright ADL, enables reasoning about the correctness of component and connector composition with respect to the interaction protocols that are used. However, from a more pragmatic standpoint, software development is greatly eased when supported by mechanisms for solving architectural mismatches, which further promotes software reuse.

Connectors that are implemented using middleware platforms actually abstract complex software systems comprising a broker, proxies, but also services for enhanced distribution management. Hence, middleware design deserves as much attention as the overall system design, and must not be treated as a minor task. Architecture-based design is again of significant assistance here. In particular, existing ADLs enable describing conveniently middleware architectures. In addition, given the fact that middleware architectures build upon well known solutions regarding the enforcement of non-functional properties, the synthesis of middleware architectures that comply with the requirements of given applications may be partly automated through a repository of known middleware architectures. In the same way, this *a priori* knowledge about middleware architectures enables one to deal with the safe dynamic evolution of the middleware architectures according to environmental changes, by exploiting both the *support for adaptation* offered by middleware platforms (*e.g.*, reflective middleware) and the *rigorous specification of software architectures* enabled by ADLs.

As briefly outlined above, results on software architectures for distributed systems primarily lie in the definition of ADLs that allow the rigorous specification of the elements composing a system architecture, which may be exploited for the system's design and, further, for the software system's assessment and construction. Work in this area also includes closer coupling with solutions that are used in practice for the development of software systems such as integration of ADLs with the now widely accepted UML standard for system modeling. Still in this direction, coupling with OMG's Model-Driven Architecture (MDA) is much beneficial. Another area that has already deserved a great deal of attention in architecture-based development is the one of easing the design and construction of middleware underpinning the system execution out of existing middleware platforms. However, addressing all the features enabled by middleware within the architecture design is not yet fully covered. For instance, this requires reasoning about the composition of, possibly interfering, middleware services enforcing distinct non-functional properties. Another area of ongoing research work from the standpoint of architecture specification relates to handling needed architectural evolution as required by emerging applications, including those based on the Internet and/or aimed at mobile computing. In this context, it is mandatory to support the development of system architectures that can adapt to the environment. As a result, the system architecture shall handle dealing with the system evolution at runtime and further assessing the behavior of the resulting system.

3.2.3. Beyond Middleware-based Architectures for Interoperability

Until now, middleware stands as the conceptual paradigm to effectively network together heterogeneous systems, specifically providing upper layer interoperability. Still, 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 now arise 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 now arising in networked environments and further called to increase dramatically in the future.

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

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

Advances in wireless networking combined with increasingly small-scale wireless devices are at the heart of the ambient intelligence (and pervasive computing) vision as they together enable ubiquitous networking and computing. However, developing software systems such that they can actually be accessed anywhere, anytime, while supporting natural interaction with users, remains a challenge. Although solutions to mobile/nomadic computing have now been investigated for more than a decade following the emergence of wireless networks and devices, these have mostly concentrated on adapting existing distributed systems architectures, so that the systems can tolerate the occurrence of disconnection. Basically, this had led to applying replication strategies to the mobile environment, where computation and/or data are cached on mobile nodes and later synchronized with peer replicas when connection allows.

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. In this context, ad hoc networking is amongst the most challenging network paradigm for distributed systems, due to its highly dynamic topology and the absence of any infrastructure. Moreover, it offers significant advantages towards the realization of ubiquitous networking and computing, still due to the absence of any infrastructure. The following section provides a brief overview of ad hoc networking and networked sensor systems, and is then followed by an overview of the key middleware functionalities that we are addressing for assisting the development of mobile systems. Such functionalities relate to the management of the network's dynamics and to enforcing system dependability.

3.3.1. Ad hoc Networking

There exist two different ways of configuring a mobile network: infrastructure-based and ad-hoc-based. The former type of network structure is the most prominent, as it is in particular used in both Wireless LANs (*e.g.*, IEEE 802.11) and global wireless networks (*e.g.*, GSM, GPRS, UMTS). An infrastructure-based wireless network uses fixed network access points (known as base stations), with which mobile terminals interact for communicating, i.e., a base station forwards messages that are sent/received by mobile terminals. One limitation of the infrastructure-based configuration is that base stations constitute bottlenecks. In addition, it requires that any mobile terminal be in the communication range of a base station. The ad-hoc-based network structure alleviates this problem by enabling mobile terminals to cooperatively form a dynamic and temporary network without any pre-existing infrastructure.

The main issue to be addressed in the design of an ad hoc (network) routing protocol is to compute an optimal communication path between any two mobile terminals. This computation must minimize the number of control messages that are exchanged among mobile terminals, in order to avoid network congestion, but also to minimize energy consumption. There exist two base types of ad hoc routing protocols: proactive and reactive. Proactive protocols update their routing table periodically. Compared to proactive protocols, reactive protocols *a priori* reduce the network load produced by the traffic of control messages, by checking the validity of, and possibly computing, the communication path between any two mobile terminals only when communication is requested between the two. Hybrid routing protocols further combine the reactive and proactive modes. The design rationale of hybrid protocols is that it is considered advantageous to accurately know only the neighbors of any mobile terminal. Since they are close to the terminal, communicating with neighbors is less expensive, and neighbors are most likely to take part in the routing of the messages sent from the terminal. Based on this, a hybrid protocol implements: (i) a proactive protocol for communication with mobile terminals in the neighborhood, and (ii) a reactive protocol for communication with the other terminals.

Spurred by the progress of technologies and deployment at low cost, the use of ad hoc networks is expected to be largely exploited for mobile computing, and no longer be restricted to specific applications (i.e., crisis applications as in military and emergency/rescue operations or disaster recovery). In particular, ad hoc networks effectively support ubiquitous networking, providing users with network access in most situations. However, we do not consider that pure ad hoc networks will be the prominent wireless networks. Instead, mobile distributed systems shall be deployed on hybrid networks, combining infrastructure-based and ad hoc networks, so as to benefit from their respective advantages. Development of distributed systems over hybrid wireless networks remains an open challenge, which requires dedicated middleware solutions for in particular managing the network's dynamics and resources, especially in the presence of multiple networks, and even individual nodes equipped with multliple radios.

3.3.2. Networked Sensor Systems

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 services 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 a deployed in 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 them.

Since their early days, WSNs have been viewed as a special class of distributed systems, and have been approached as such from an application development perspective as well. Consequently, application developers have thus far specified their applications at the level of the individual node where the they use a language such as nesC, galsC or Java to write the program, directly interacting with the node-level services stated earlier, or a middleware that aids in the programming process. The developer can read the values from local sensing interfaces, maintain application level state in the local memory, send messages to other nodes addressed by node ID or location, and process incoming messages from other nodes. However, In all these approaches. the application developer is responsible for ensuring that these individual finite state-machines executing on the individual nodes of the WSN will interact to produce the desired result. Owing to the large size and heterogeneity of the systems involved, as well as the limited distributed programming expertize of the domain experts, the above paradigm of node-level programming is not easy to use for sensor networks. We believe that this is a large obstacle holding back the wide-acceptance of WSNs. For example, to develop an environment management application in nesC, a commonly used language in WSNs, the developer has to specify the functions at each node in terms of the respective components - one each for sensing the environment, communicating with other nodes, as well as controlling the actuators attached to each nodes. In addition to the above, the application developer is also responsible for ensuring that the distributed application that results from these communicating node-level programs performs the necessary functions as desired, and is also efficient in terms of the energy spent during its operations.

Sensor network *macroprogramming* aims to aid the wide adoption of networked sensing by providing the domain expert the ability to specify their applications at a high level of abstraction. In macroprogramming, abstractions are provided to specify the high-level collaborative behavior at the system level, while intentionally hiding most of the low-level details concerning state maintenance or message passing from the programmer. As a result of this, macroprogramming is emerging as a viable technique for developing complex embedded applications, as demonstrated by the several efforts currently underway in this field.

The challenges we focus on as part of our work on networked sensor systems are twofold - suitable macroprogramming abstractions for enabling easy development of sensor network applications, and supporting these techniques by developing suitable underlying middleware to tie together the increasingly heterogeneous systems.

3.3.3. Managing the Network's Dynamics

Trends in mobile computing have created new requirements for automatic configuration and reconfiguration of networked devices and services. This has led to a variety of protocols for lookup and discovery of networked resources. In particular, *discovery protocols* provide proactive mechanisms for dynamically discovering, selecting and accessing available resources. As such, resource discovery protocols constitute a core middleware functionality towards managing the network's dynamics in mobile computing systems. Resource discovery is a central component of distributed systems as it enables services and resources to discover each other on a network and evaluate potential interactions. Many academic and industry-supported protocols (*e.g.*, SLP, UDDI, SSDP) have been designed in different settings, and numerous are now in common usage, using either distributed or centralized approaches depending on assumptions about the underlying network and the environment. These design constraints have led to different, sometimes incompatible mechanisms for service advertisements, queries, security and/or access, while none of the existing resource discovery protocols is suitable for all environments.

The major structural difference between existing resource discovery protocols is the reliance (or not) on a central directory. A central directory stores all the information concerning resources available in the network, provided that resources advertise themselves to the central directory using a unicast message. Then, to access a resource, a client first contacts the central directory to obtain the resource's description, which is to be used for contacting the resource's provider. Prior to any resource registration or client request to the central directory, clients and resource providers must first discover the central directory by issuing broadcast or multicast requests. Centralized resource discovery is much suited to wireless infrastructure-based networks. However, this makes the discovery process dependent upon the availability of the central directory, which further constitutes a bottleneck. In order to support resource discovery in a wider network area, the use of a distributed set of fixed directories has been proposed. Directories are deployed on base stations (or gateways) and each one is responsible for a given discovery domain (*e.g.*, corresponding to a cell).

In the self-organizing wireless network model provided by ad hoc networks that use peer-to-peer communication and no fixed infrastructure, the use of fixed directories for resource discovery is no longer suitable. In particular, the selection of mobile terminals for hosting directories within an ad hoc network is a difficult task, since the network's topology frequently changes, and hence the connectivity is highly dynamic. Decentralized resource discovery protocols then appear more suitable for ad hoc networks. In this case, resource providers and clients discover each other directly, without interacting with a central directory. Specifically, when a client wants to access a resource, it sends a request to available providers using a broadcast message. However, this approach leads to the flooding of the network. An approach to disseminating information about network resources while not relying on the use of broadcast is to use geographic information for routing. Nodes periodically send advertisement along a geometric trajectory (basically north-south and west-east), and nodes located on the trajectory both cache and forward advertisements. Then, when a client seeks a resource, it sends a query that eventually intersects an advertisement path at a node that replies to the request. This solution assumes that the density of nodes is high enough, and further requires the replication of resource advertisements on a significant number of nodes. Hence, it incurs resource consumption that may not be accommodated by wireless, resource-constrained nodes. Resource consumption is further increased by the required support for geographical location (*e.g.*, GPS). Other solutions to decentralized resource discovery that try to minimize network flooding are based on local resource discovery. Broadcast is limited to the neighborhood, hence allowing only for resource discovery in the local area, as supported by base centralized resource discovery protocols. Discovery in the wider area then exploits solutions based on a hierarchy of discovery domains.

Resource discovery protocols for hybrid networks that in particular suit ad hoc networks remains an open issue. Other fundamental limitations of the leading resource discovery protocols are: (i) reliance on syntactic matching of resource attributes included in the resource description, and (ii) unawareness of the environment where the resources are provided. The development of mobile/handheld devices, and wireless and ad hoc networks (*e.g.*, Wi-Fi, Bluetooth) have enabled the emergence of service-rich environments aimed at supporting users in their daily life. In these pervasive environments, a variety of infrastructure-based and/or infrastructure-less networks are available to the users at a location. Such heterogeneous environments bring new challenges to resource/service discovery, including context-awareness, protocol interoperability, semantic knowledge, etc.

While resource discovery constitutes a core middleware functionality towards easing the development of distributed software systems on top of dynamic networks, higher-level abstractions for dynamic networks need to be developed and supported by the middleware for easing the developers' task. The definition of such abstractions shall be derived from both features of the network and architectural principles elicited for mobile software systems, where we exploit our work in both areas. Related issues include characterizing and reasoning about the functional and non-functional behavior of the participating peer nodes, and in particular dealing with security requirements and resource availability that are crucial in the mobile environment.

3.3.4. Enforcing Dependability

Dependability of a system is defined as the reliance that can justifiably be placed on the service that the system delivers. It decomposes into properties of availability, safety, reliability, confidentiality, integrity and maintainability, with security encompassing availability, confidentiality and integrity. Dependability affects the overall development process, combining four basic means that are fault prevention, fault removal, fault tolerance and fault forecasting. In the context of middleware architectures for mobile systems, we concentrate more specifically on fault tolerance means towards handling mobility-induced failures. Such failures affect most dependability properties. However, availability and security-related properties are the most impacted by the mobile environment due to changing connectivity and features of wireless networks that make them more prone to attacks. Security remains one of the key challenges for mobile distributed systems. In particular, the exploitation of ad hoc networks does not allow systematic reliance on a central infrastructure for securing the network, calling for decentralized trust management. Additionally, resource constraints of mobile devices necessitate the design of adequate cryptographic protocols to minimize associated computation and communication costs.

Enforcing availability in the mobile environment relies on adequate replication management so that data and/or services remain accessible despite the occurrence of disconnection. Such a concern has led to tremendous research work since the emergence of mobile computing. In particular, data replication over mobile nodes has led to novel coherency mechanisms adapted to the specifics of wireless networks. Solutions in the area relate to offering optimistic coherency protocols, so that data copies may be concurrently updated and later synchronized, when connectivity allows. In initial proposals, data copies were created locally on accessing nodes, since these proposals were aimed at global infrastructure-based networks, where the mobile node either has access to the data server or is isolated. However, today's wireless networks and in particular ad hoc networks allow for creating temporary collaborative networks, where peer nodes may share resources, provided they trust each other. Hence, this allows addressing the replication of data and services over mobile nodes in accordance with their respective capabilities. Dually, peer-to-peer communication supported by ad hoc networks combined with decentralized resource discovery allow accessing various instances of a given resource, and hence may be conveniently exploited towards increasing availability. Today's wireless networks offer great opportunities towards availability management in mobile systems. However, providing effective solutions remains an open issue, as this must be addressed in a way that accounts for the constraints of the environment, including possible resource constraints of mobile nodes and changing network topology. Additionally, solutions based on resource sharing among mobile nodes require incentive mechanisms to avoid selfish behavior where nodes are trying to gain but not provide resource access.

4. Application Domains

4.1. Application Domains

The ARLES project-team targets development support for applications relevant to the ambient intelligence/pervasive computing vision, with a special focus on consumer-oriented applications. Architecture-based development of composite systems is further directly relevant to enterprise information systems, whose composition is mainly static and relates to the integration of legacy systems. In addition, by building upon the Web services architecture for dealing with the dynamic composition of (possibly mobile) autonomous systems, our work is of direct relevance to e-business applications, providing specific solutions for the mobile context. Finally, we also address the issues brought into focus by the increased presence of wireless sensors and actuators in contexts including home and office environments.

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 & 7.3).

As an illustration of applications investigated within ARLES, the COCOA semantic service middleware together with the INMIDIO interoperable middleware (respectively, § 5.6 & 5.5) support the *networked home* environment. The networked home specifically seeks to combine the home automation, consumer electronics, mobile communications and personal computing domains to provide new user applications that exploit the fluid integration of these traditionally strictly separated domains, and to lay a solid foundation towards realizing the ambient intelligence vision. The iBICOOP middleware (§ 5.8) enables mobile data management in multi-radio/multi-network/multi-device scenarios, and the *Srijan* toolkit (§ 5.9) aids the development of complex applications on heterogeneous networked sensor systems at a high level.

5. Software

5.1. Introduction

In order to validate our research results, our research activities encompass the development of related prototypes. We present here chronologically the prototypes that are released under open source license at http://www-rocq.inria.fr/arles/doc/doc.html.

5.2. WSAMI: A Middleware Based on Web Services for Ambient Intelligence

Participant: Valérie Issarny [correspondent].

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 [7]. WSAMI further supports the dynamic composition of distributed services over hybrid wireless networks. Moreover, WSAMI takes in charge the customization of the network's path through the dynamic integration of middleware-related services, in order to enforce quality of service with respect to offered dependability and performance properties.

The WSAMI middleware prototype is available since 2004. It is a Java-based implementation of the WSAMI core middleware, which builds upon IEEE 802.11b as the underlying WLAN and integrates the following components: (i) the WSAMI SOAP-based core broker, including the CSOAP[6] SOAP container for wireless, resource-constrained devices; and (ii) the Naming & Discovery service, including support for connector customization. The memory footprint of our CSOAP implementation is 90kB, as opposed to the 1100kB of the Sun's reference SOAP implementation. The overall memory footprint of our Web services platform is 3.9MB, dividing into 3MB for the CVM and 815kB for the Xerces XML parser, in addition to the CSOAP implementation. The WSAMI middleware prototype is an open-source software freely distributed since 2004 under the terms of the GNU Lesser Public License (LGPL) at http://www-rocq.inria.fr/arles/download/ozone/index.htm.

Our prototype is being used for the implementation of demonstrator applications in the field of ambient intelligence, as well as a core for service-oriented middleware platforms aimed at advanced wireless networking environments, like Ariadne, presented below.

5.3. Ariadne: A Protocol for Scalable Service Discovery in MANETs

Participant: Valérie Issarny [correspondent].

The Ariadne service discovery protocol has been designed to support decentralized Web service discovery in multi-hop mobile ad hoc networks (MANETs) [10]. Ariadne enables small and resource-constrained mobile devices to seek and find complementary, possibly mobile, Web services needed to complete specified tasks, while minimizing the traffic generated and tolerating intermittent connectivity. The Ariadne protocol further enables service requesters to differentiate services instances according to non-functional properties. Specifically, the Ariadne protocol is based on the homogeneous and dynamic deployment of cooperating directories within the MANET. Scalability is achieved by limiting the generated traffic related to service discovery, and by using compact directory summaries (i.e., Bloom filters) to efficiently locate the directory that most likely has the description of a given service.

The prototype of the Ariadne service discovery protocol has been implemented in Java, and provides an application programming interface (API) so as to be easily integrated in a Web service-oriented middleware such as WSAMI, presented above. The Ariadne prototype is an open source software freely distributed since 2005 under the terms of the GNU Lesser Public License (LGPL) at http://www-rocq.inria.fr/arles/download/ariadne/index.html.

Our prototype is being used for the implementation of demonstrator applications exploiting MANETs in the field of ambient intelligence.

5.4. MUSDAC: A Middleware for Service Discovery and Access in Pervasive Networks

Participant: Valérie Issarny [correspondent].

[6]Compact SOAP

The MUlti-protocol Service Discovery and ACcess (MUSDAC) middleware platform enables the discovery and access to services in the pervasive environment, which is viewed as a loose and dynamic composition of independent networks [8]. 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.

We have implemented a first prototype of MUSDAC in Java (J2SE 1.4.2 and 1.5), which includes support for 5 different service discovery protocols, and remote access for SOAP-based services. The different plug-ins enable us to experiment with both repository-based (Ariadne, OSGi) and multicast-based (SLP, UPnP) protocols. The MUSDAC prototype is an open source software freely distributed since 2005 under the terms of the GNU Lesser Public License (LGPL) at http://www-rocq.inria.fr/arles/download/ubisec/index.html.

The MUSDAC middleware in particular serves as a base building block in the development of a middleware aimed at effectively enabling service-oriented computing in Beyond 3rd Generation (B3G) networks. This further led to make evolve MUSDAC-based service discovery and access to multi-radio and multi-protocol networking environments (see § 5.7).

5.5. INMIDIO: An Interoperable Middleware for Ambient Intelligence

Participant: Nikolaos Georgantas [correspondent].

In the pervasive computing environment, devices from various application domains, *e.g.*, home automation, consumer electronics, mobile and personal computing domains, need to dynamically interoperate irrespective of the heterogeneity of their underlying hardware and software. Middleware has been introduced in order to overcome this issue by specifying a reference interaction protocol enabling compliant software systems to interoperate. However the emergence of different middleware platforms to address the requirements of specific application domains leads to a new heterogeneity issue among interaction protocols. Thus, at a given time and/or at a specific place, devices hosting the wrong middleware become isolated. In order to overcome this issue, we have developed a system called INMIDIO (INteroperable MIddleware for service Discovery and service InteractiOn) that 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 [3]. Specifically, the system parses the incoming/outgoing message and, after having interpreted the semantics of the message, it generates a list of semantic events and uses this list to reconstruct a message for the target protocol, matching the semantics of the original message. The INMIDIO middleware acts in a transparent way with regard to discovery and interaction middleware protocols and with regard to the services running on top of them.

The service discovery protocols supported by the current INMIDIO prototype are UPnP, SLP and WS-Discovery, while the supported service interaction protocols are SOAP and RMI. The INMIDIO prototype is publicly available since 2006 and released under the GNU Lesser Public License (LGPL) at http://www-rocq.inria.fr/arles/download/inmidio/index.html.

5.6. COCOA: A Semantic Service Middleware

Participant: Nikolaos Georgantas [correspondent].

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 [1], [2]. Using COCOA, abstract user tasks are realized by dynamically composing the capabilities of services that are currently available in the environment. The capabilities that a service provides are presented as a *conversation* – a workflow that specifies data and control dependencies for its capabilities. Similarly, an abstract task is presented as an orchestration of required capabilities. The conversations of the provided service capabilities are integrated to realize the orchestration of the abstract task, while guaranteeing that the dependencies of each of the provided capabilities are preserved. This allows complex user tasks to be created and reliably composed, while offering fine-grained control over the placement of capabilities in the task. This is especially important for the pervasive environment, where user tasks may frequently involve interaction with the user(s). In addition, the service composition can be optimized based on quality of service and context-aware parameters. To accommodate the inherent heterogeneity of services in the pervasive environment, capabilities are described and matched semantically, and adapted when necessary. Furthermore, different service groundings are supported, allowing diverse SOA platforms to be incorporated; interoperability at service grounding, *i.e.*, middleware level may then be ensured by employing INMIDIO (§ 5.5). Once a new service realizing the user task has been created, it is automatically deployed and executed.

A prototype version of COCOA has been released under LGPL as open source software in 2007 on the Amigo GForge Open Source Software site http://gforge.inria.fr/projects/amigo/.

5.7. ubiSOAP: A Service Oriented Middleware for Seamless Networking

Participant: Valérie Issarny [correspondent].

The ubiSOAP middleware empowers the service-oriented architecture with pervasive networking capabilities, in particular enabling adaptive lightweight services to be run on mobile nodes and access to services over multi-radio, multi-network links [4].

The ubiSOAP middleware specifically enriches the Web service architecture with key features for services to become truly pervasive by taking full benefit of the rich capacities, including multi-radio interfaces, now embedded in wireless devices. Specifically, ubiSOAP brings multi-radio, multi-network connectivity to services through a comprehensive layered architecture:

- 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.
- The communication layer allows for communication in the pervasive networking environment according to SOAP. ubiSOAP in particular enriches traditional functionalities of a SOAP engine to allow for SOAP-based point-to-point and group-based interactions in the pervasive network. It further enables access to services that may be in distinct networks thanks to multi-network routing.
- The middleware services layer brings advanced distributed resource management functionalities customized for the pervasive networking environment. From that layer, the Discovery Service enables the dynamic advertising and location of networked services, in particular accounting for extra-functional properties.

Last but not least, the ubiSOAP middleware is mobility-aware so that its functionalities adapt to the physical mobility of both clients and services, in particular exploiting the rich multi-radio, multi-network connectivity.

The ubiSOAP middleware has been implemented using Java for both desktop (J2SE) and mobile (J2ME CDC) environments. The software has been developed in the context of the IST PLASTIC Project and can be freely downloaded since 2008 under the GNU LGPL license at http://www.ist-plastic.org.

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

Participant: Valérie Issarny [correspondent].

To better support interactions between mobile users, we are developing the iBICOOP middleware [18]. Our 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 up-coming multi-device, multi-user world.

- *Communication Manager:* The iBICOOP Communication Manager is aimed to overcome the constraints of different network characteristics that different devices may have. 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 communication between two devices is always secured with SSL.
- *Security Manager:* The iBICOOP Security Manager uses well-established techniques of cryptography and secure communication to provide necessary security. Presently, we are providing RSA, AES, DES, and Diffie-Hellman for generating keys that can be used for encryption/decryption of data for storing on device or sending over the net. The Security Manager is also responsible for access control on shared devices.
- *Partnership Manager:* The iBICOOP Partnership Manager provides device or user information in the form of *profiles*. Profiles are stored as XML files. This allows us to easily integrate our profiles with the IMS architecture using XDMS (XML Data Management Server). We have defined XML schema for profile. In the iBICOOP middleware, profiles are used by core modules to make a service available for user. To make our solution privacy-aware, the Partnership Manager module provides filters to control the information exchanged in a profile.
- *Naming & Discovery:* iBICOOP's naming scheme is based on hierarchical names in the pattern of URI scheme http://tools.ietf.org/html/rfc3986; We combine names of protocol, devices, services, email etc. to give a human-readable name to a resource. An example for a service is: "ibic://john@work.com:ipaqpda/axis2/services/exchange". For iBICOOP, we rely on Service Location Protocols to find nearby services on currently active network interfaces that support IP multicast. Setting up partnerships between users (bootstrapping a relationship) may also use the discovery service, and may happen after a multicast-based local discovery (e.g., over WiFi) for any Partnership Manager service, or by searching for such services on a public repository.
- *Local File Manager:* Besides normal file managing tasks, the iBICOOP Local File Manager gives user clear cues to the files that have been replicated across multiple devices or shared among different users by using different icons. Later it will provide extra functionalities for content-sharing and data management in emerging pervasive computing environments

More details about the current status and future work on iBICOOP is in § 6.4

5.9. Srijan: Data-driven Macroprogramming for Hetergeneous 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. In this research, 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 autogenerated 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. The software has been released as open source at https://gforge.inria.fr/projects/srijan/.

6. New Results

6.1. New Results

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. We in particular study ambient intelligence system architectures based on the service paradigm, from service modeling to service-oriented middleware for advanced wireless networks. Additionally, this year we have been initiating research into two directions – dynamic interoperability between networked systems, and high-level application development in sensor networks – both called for by the ongoing drastic evolution of the networking environment. Our research activities over the year 2009 have focused on the following complementary areas:

- Maintenance of Service-Oriented Software, by building on lessons learnt in the Object-Oriented Programming community (§ 6.2),
- QoS-aware Service-Oriented Middleware for Pervasive Environments(§ 6.3),
- Middleware support to allow ubiquitous access of user data from a multitude of devices with heterogeneous capabilities and running on different platforms (§ 6.4).
- Dynamic interoperability among networked systems towards making them eternal, by way of on-the-fly generation of connectors based on adequate system models (§ 6.5),
- Enabling End-User Composition of Pervasive Applications, to make pervasive computing applications accessible to the masses (§ 6.6), and
- High-level application development in heterogeneous sensor networks, focussing specifically on data-driven macroprogramming (§ 6.7).

6.2. Maintenance of Service Oriented Software

Participants: Dionysis Athanasopoulos, Apostolos Zarras, Valérie Issarny.

Service-Oriented Architecture (SOA) is an architectural style that emerged as the answer to the latest requirements for loosely-coupled distributed computing. Inline with the conventional distributed computing paradigm, functionality is decomposed into distinct architectural elements, distributed over the network. Nevertheless, in SOA the basic architectural elements (a.k.a services) are by themselves autonomous systems that have been developed independently from each other. We worked on two areas in this domain in 2009.

Supporting Maintainability in Service-oriented Software. Until now, state of the art research in SOA systems has focused mostly on issues concerning the construction phase of service-oriented software. The outcome of these research efforts was mechanisms for discovering, composing and accessing available services. However, several other phases of the development process are currently underdeveloped. In [15], we focus on the *maintenance phase of service-oriented software*, i.e., software built by composing services. Specifically, we concentrate on the *maintainability quality attribute*. The importance of this issue is evident towards the success of the SOA paradigm, which promotes the development of software consisting of independently evolving basic engineering elements that may further vary in quality (e.g., performance, availability, reliability).

In conventional Object-Oriented (OO) software, maintainability can be improved by employing well known fundamental design principles such as OCP (Open Closed Principle), DIP (Dependency Inversion Principle) and LSP (Liskov Substitution Principle). We revisited these principles in the context of the SOA paradigm and argue about the need to adapt/refine them to the specificities that characterize the paradigm. Specifically, we:

- examined the maintenance scenarios that can be handled by the conventional use of the fundamental design principles in the SOA paradigm and discuss why these scenarios are not realistic,
- adapted/refined the fundamental design principles such that their use in service-oriented software becomes effective towards handling realistic maintenance scenarios, and
- sketched the ForeverSOA infrastructure, which aims at facilitating the adoption of the refined principles in the development of SOA software. The prominent concept in ForeverSOA is a reverse engineering process that recovers service abstractions out of available services. An abstraction characterizes a group of services, providing similar functionalities via different interfaces and serves for developing software that may access any of the grouped services without depending on their interfaces.

Reducing the Complexity of Service Substitution. Service substitution is a key issue towards dealing with the independent evolution of services along with their variation in quality (e.g., performance, availability, reliability). Research efforts that focus on service substitution can be divided in two categories. The first category consists of *abstraction-based* approaches that propose development methodologies and frameworks that allow the *development from scratch of client applications, which use service abstractions that are mapped into alternative concrete services*. The second category comprises of adapter-based approaches, which deal with existing client applications that use concrete services. The basic concept in this case is to derive a *mapping* between the target service that should be substituted and a *substitute service* that offers similar functionality through a different interface. Based on such a mapping, an adapter is generated, which allows accessing the functionality of the substitute service through the original target interface, without modifying the client application code.

While considering adapter-based approaches the following issue is raised: the effort and time required by the service substitution process scales up with the number of available services that should be examined as potential candidate substitutes of the target service. This problem is a serious drawback towards a practical service substitution approach if we consider that the service substitution process involves human intervention to validate the mapping between target and substitute services.

Our work shares the objective of adapter-based approaches. However, our specific goal is to reduce the effort and time required to achieve this objective. To this end, we propose a hybrid approach [16] that borrows ideas from abstraction-based approaches so as to handle the complexity of service substitution. The proposed approach relies on a formal foundation that comprises two substitution relations and corresponding substitution theorems, which are in line with the Liskov substitution principle (LSP). Based on the proposed relations and theorems, available services are organized into groups, characterized by abstractions, called profiles. Then, the complexity of service substitution scales up with the number of available profiles, instead of scaling up with the number of available services. Our experimental results highlight the aforementioned benefit. Currently, we are working towards a reverse engineering process that would allow to improve the organization of services into groups, by recovering service abstractions from a set of available services. The proposed framework may be extended to account for mismatches in the order of operations; it may also be combined with keywords-based and QoS-based search techniques.

6.3. QoS-aware Service-Oriented Middleware for Pervasive Environments

Participants: Nebil Ben Mabrouk, Nikolaos Georgantas, Sandrine Beauche.

Pervasive computing has changed the role, the characteristics and the requirements of software supporting users in their daily life. Hence, the way software systems are designed and built must change accordingly. A specific feature to consider towards this purpose is Quality of Service (QoS). Indeed, pervasive computing brings to the fore new challenges and issues related to QoS.

As part of our research, we investigate a middleware-based solution addressing QoS concerns in pervasive computing. The proposed solution copes with QoS issues related to different phases of the pervasive systems' life cycle. This includes (i) QoS description during the specification of pervasive systems, (ii) QoS-awareness during the design and implementation of pervasive systems, and (iii) the adaptation of these systems regarding QoS dynamics.

In the following paragraphs, we detail the current status of our solution. The two first requirements have been addressed (i.e., QoS specification and QoS-aware design and implementation of pervasive systems), whereas preliminary investigations about QoS-driven adaptation for pervasive systems have been conducted.

A Semantic End-to-End QoS Model for Pervasive Computing. Addressing QoS concerns in pervasive computing requires defining a QoS model that provides the appropriate ground for QoS specification and provision in pervasive environments. Towards this purpose, we propose a semantic QoS model [22] that addresses QoS on an end-to-end basis by covering quality factors of the main elements acting in pervasive environments, in particular it deals with: (i) network and hardware resources, (ii) application services and (iii) end-users. Our model puts a special emphasis on QoS features related to the dynamic nature of these resources such as end-user mobility and context awareness of application services. Our model is designed according to a layered approach integrating four QoS ontologies: (1) *QoS Core ontology*, (2) *Infrastructure QoS ontology*, (3) *Service QoS ontology* and (4) *User QoS ontology*.

The proposed ontologies focus on representing QoS knowledge with rich semantic information rather than specifying a language for QoS. Coupled with XML-based QoS specifications, these ontologies can formulate a robust QoS description framework that combines the rich semantics of QoS ontologies with the accuracy of QoS specification languages. The proposed ontologies provide support to the remaining parts of our middleware solution addressing QoS concerns in pervasive environments, namely for the QoS-aware service composition which will be detailed in the next paragraph.

A QoS-aware Service Composition Algorithm for Pervasive Systems. QoS-aware service composition is a key requirement in pervasive computing since it enables selecting services able to fulfill complex user tasks while meeting their QoS requirements. Nevertheless, the highly dynamic nature of pervasive environments raises several challenges regarding QoS-aware service composition. To cope with these challenges we propose a heuristic algorithm [23] for QoS-aware service selection and composition on the fly.

The importance of our algorithm is three-fold. First, it introduces a novel approach based on clustering techniques. Applying such techniques for services' selection brings new ideas to this research area. Second, by producing not a single but multiple service compositions our algorithm allows for coping with changing conditions in pervasive environments. This approach provides support to many features required for pervasive systems such as dependability and self-adaptation. Third and most importantly, our algorithm shows a satisfying efficiency in terms of timeliness and optimality, which makes it appropriate for on-the-fly service composition in pervasive environments. A set of experiments have been conducted to validate our algorithm.

QoS-driven Self-adaptation Approach for Pervasive Systems. Addressing QoS concerns during the design and the implementation of pervasive systems is not enough to ensure steadily a satisfying level of QoS since QoS can change dynamically at runtime. To this regard, pervasive systems need to adapt themselves according to QoS dynamics. Towards this purpose, we present an approach for QoS-driven self-adaptation of pervasive systems at runtime. Our approach is based on two strategies of self-adaptation. First, runtime self-adaptation at the system level (i.e., the system implementation) based on the substitution of services composing the system dynamically at runtime. This approach is underpinned by our QoS-aware selection algorithm allowing dynamic binding of services.

If the substitution fails (e.g., there are no substituting services), we bring up self-adaptation of the system to the architectural level (i.e., the system design). This second strategy aims at finding an alternative architecture able to fulfill the user task. It is based on the concept of *Task Class* that we introduce to define the set of abstract composition specifications which are functionally equivalent (i.e., allowing to achieve the same task). Based on the Task Class concept, our approach allows for adapting pervasive systems by switching from the current composition specification (i.e., associated with the running system) to an alternative one allowing to achieve the same goal.

The two self-adaptation strategies (i.e., at the system level and the architecture level) are coordinated at runtime by continuously evaluating the status of the system.

6.4. Data Sharing and Replication in Pervasive Networks

Participants: Roberto Speicys Cardoso, Sneha Godbole, Pushpendra Singh, Amel Bennaceur, Valérie Issarny.

Nowadays, users rely on various sets of connected devices and services to interact with each other: home gateways, laptops, smartphones, UMPC, enterprise servers, Web/cloud computing services, etc. While for a long time interactions have been mostly Web-centric, users now routinely engage in user centric interactions such as content sharing and social networking through mobile devices. Still, while mobile devices have become very capable of communicating with other devices, accessing or exchanging content is far from being easy: communication costs are still high, communication protocols are not uniformly supported, communication links are not always reliable, etc. Another problem lies in the role usually assigned to mobile devices. Although mobile devices are the primary terminals for interaction, they have essentially focused on (i) acting as clients accessing services in the infrastructure (e.g., enablers in the IMS infrastructure for Telecoms networks, or Web services in the Internet), or (ii) manipulating content stored locally (e.g., multimedia content playback). So far, there has been little success in truly integrating these mobile devices in the pervasive computing environment (i.e., acting also as resource or service providers). Furthermore, the use of multiple devices, having large storage capabilities, has resulted in the scattering of content on a set of devices, which, ironically, is making access to all the content that one possesses a big challenge (Where is this file? Is it the latest version?). This problem becomes acute when it comes to impromptu collaboration (e.g., in business meetings, conferences, or on the road). Such collaborations often require exchange of content, and have to wait till the user gets physical access to the device over which the required content (or the correct/latest version) is currently stored. Delay in accessing the content often cause frustration and missed opportunities.

To answer these challenges, and better support interactions between mobile users, we are developing the iBICOOP middleware [18]. Our middleware addresses these challenges 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 up-coming multi-device, multi-user world.

- *Communication Manager:* The iBICOOP Communication Manager is aimed to overcome the constraints of different network characteristics that different devices may have. 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 communication between two devices is always secured with SSL.
- *Security Manager:* The iBICOOP Security Manager uses well-established techniques of cryptography and secure communication to provide necessary security. Presently, we are providing RSA, AES, DES, and Diffie-Hellman for generating keys that can be used for encryption/decryption of data for storing on device or sending over the net. The Security Manager is also responsible for access control on shared devices.
- *Partnership Manager:* The iBICOOP Partnership Manager provides device or user information in the form of *profiles*. Profiles are stored as XML files. This allows us to easily integrate our profiles with the IMS architecture using XDMS (XML Data Management Server). We have defined XML schema for profile. In the iBICOOP middleware, profiles are used by core modules to make a service available for user. To make our solution privacy-aware, the Partnership Manager module provides filters to control the information exchanged in a profile.

- *Naming & Discovery:* iBICOOP's naming scheme is based on hierarchical names in the pattern of URI scheme http://tools.ietf.org/html/rfc3986; We combine names of protocol, devices, services, email etc. to give a human-readable name to a resource. An example for a service is: "ibic://john@work.com:ipaqpda/axis2/services/exchange". For iBICOOP, we rely on Service Location Protocols to find nearby services on currently active network interfaces that support IP multicast. Setting up partnerships between users (bootstrapping a relationship) may also use the discovery service, and may happen after a multicast-based local discovery (e.g., over WiFi) for any Partnership Manager service, or by searching for such services on a public repository.
- *Local File Manager:* Besides normal file managing tasks, the iBICOOP Local File Manager gives user clear cues to the files that have been replicated across multiple devices or shared among different users by using different icons. Later it will provide extra functionalities for content-sharing and data management in emerging pervasive computing environments

We are currently implementing the iBICOOP middleware in Java, using IBM J9's JVM with CDC 1.1 for Windows mobile PDA and smartphones, as well as native CLDC/MIDP for other phones and smartphones. On the infrastructure side, services (e.g., Communication proxy service and Discovery service) are developed in Java 1.6, and deployed on Apache Tomcat and on the Alcatel 5350 IMS application server (for the IMS infrastructure). A number of core modules have already been implemented (Partnership Manager, Security Manager, and Communication Manager) on specific devices and are being ported to C# (Windows Mobile) and Objective-C (iPhone). We plan to implement iBICOOP as a tight integration between services on the Internet, local networks, and in the IMS architecture thus providing a solution for both the Telecoms and Internet world.

We are also building two collaboration services on iBICOOP core modules: the *Replication Service* and the *Exchange & Sharing service*.

- *Replication Service:* The iBICOOP Replication Service keeps track of the files that are replicated over the user workspace. A workspace is the total space available on all the user devices. Users are made aware of conflicts if they try to synchronize with a copy that has been modified on another device. Users can add or remove files from replication service when they wish so. A multi-criteria algorithm is used to choose the best way to transfer data during replication.
- *Exchange & Sharing Service:* The Exchange and Sharing Service takes the abstraction to a higher level by allowing several users to interact and collaborate in various scenarios. No matter where data is located, users can transfer and share data between their workspaces. The aim of this service is to make long-term collaborations or impromptu exchanges an easier and secure task.

In iBICOOP, we are developing a middleware to allow ubiquitous access of user data from a multitude of devices with heterogeneous capabilities and running on different platforms. The services that we are building on top of iBICOOP aim to show the viability of iBICOOP as a standard-based platform for future advance services. Leveraging Telecoms and Internet world, integration with IMS architecture, is one of the salient feature of iBICOOP. The iBICOOP middleware offers a complete solution for end-user with regards to content-sharing and data management in emerging pervasive computing environments.

6.5. Dynamic Synthesis of Middleware Connectors

Participants: Romina Spalazzese, Valérie Issarny.

Interoperable middleware have been introduced to overcome middleware heterogeneity. However, the solutions remain rather static, requiring either the use of a proprietary interface or a priori implementation of protocol translators. In general, interoperability solutions solve protocol mismatch among middleware at the syntactic level, which is too restrictive. This is even truer when one considers the many dimensions of heterogeneity, including software, hardware and networks, which now arise 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 now arising in networked environments and which will also increase dramatically in the future.

Mediator then stands as a core architectural paradigm for today's and future systems that increasingly need be connected. The mediator concept was early introduced to cope with the integration of heterogeneous data sources. However, with the significant development of Web technologies and given abilities to communicate openly for networked systems, many heterogeneity dimensions shall now be mediated among which protocol mediation that is concerned with behavioral mismatches that may occur during interactions.

A key challenge for today's systems architectures is to embed the necessary support for automated mediation, i.e., the connector concept needs to evolve towards the one of mediator connectors. Indeed, the actual systems with which communications will take place cannot be anticipated at design time due to today's open networking and further continuous evolution of networked systems. As such, connectors not solely coordinate the interaction behaviors of connected systems but also mediate those behaviors to enable actual interactions.

Automated mediation of heterogeneous protocols basically relies on: (i) the adequate modeling of processes abstracting the behavior of the protocols to be bridged (ii) the definition of a matching relationship between the process models that sets the conditions under which protocol interoperability is supported (iii) the elicitation of an algorithm that computes an appropriate mapping between matching process models.

A number of solutions to automated protocol mediation have recently emerged. Although proposed algorithms manipulates formally grounded process models, most solutions are discussed informally, making it difficult to assess their respective advantages and drawbacks. They further remain rather vague on the definition of enforced matching relationship.

What is needed is a new and formal foundation for mediator connectors from which protocol matching and associated mapping relationships may be rigorously defined and assessed. These relationships may be automatically reasoned upon, thus paving the way for on the fly synthesis of mediator connectors. To the best of our knowledge such an effort has not been addressed in the past.

The problem of protocols mediation concerns the interoperability between two protocols. To explain the problem, let us consider for example two instant messengers: Windows Messenger(W) and Jabber Messenger(J). With respect to protocol meditation we are interested to formalize: i) how to identify whether the messenger protocols share similar intent ii) how to synthesise the mediator connector that enables them to interoperate despite of mismatches. To be precise, by "protocols share similar intent" we mean that given the interaction protocols P_W and P_J of W and J respectively, part of their behavior is complementary thus showing an interaction potentiality. Thus, we expect to find, at a given level of abstraction, similarities in the structure of the protocol representations. This leads to formally analyze such alike protocols to find, if it exists, a suitable mediator that allows the interoperability that otherwise would not be possible.

As part of our work in the CONNECT project, we have introduced a first formal characterization of the mediating connector concept in [25]. The proposed description paves the way for automated reasoning about protocol matching and mapping and thus for on the fly synthesis of mediating connectors. We assume to know the behavioral specifications B_W and B_J of the protocols P_W and P_J respectively and also to have an ontology mapping between the alphabets of the mismatching protocols that defines a common alphabet. The high level process that we are investigating to solve this problem is made by the following steps. The first one deals with the transformation of B_W and B_J into canonical forms, CB_W and CB_J respectively, in order to simplify them and reason on minimized protocols. The second step deals with the check of the compatibility between CB_W and CB_J. If they are compatible then one tries to synthesise the mediator connector to allow P_W and P_J to interoperate otherwise the interoperability is not possible.

The compatibility check (also called matching), if succeeds, singles out the parts of the structures of the protocols that are expected to be similar, i.e., the parts of actual communication between P_W and P_J . Then a mapping will build the behavioral representation for the abstract mediator for such part of protocols. The subsequent synthesis produces the actual implementation of the mediator. To validate the proposed approach, the protocols being rigorously defined, the correctness and the completeness of the synthesised mediator can be formally proved.

The approach is a first attempt, it only partly covers the existing mismatches, and needs to be extended to cover a larger set. Future work concerns several investigation areas among which an assessment of the generality of the proposed framework and the realization of semi-automatic support to the elicitation and the subsequent synthesis of the mediator.

6.6. Enabling End-User Composition of Pervasive Applications

Participants: Oleg Davidyuk, Nikolaos Georgantas, Valérie Issarny.

The vision of ubiquitous computing emphasizes user interaction with public and private spaces that consist of humans and multiple interconnected computing resources. This vision has been recently motivated further by the increasing number of embedded computing resources that use wireless communication. As a consequence, users nowadays can access services anywhere and interact with surrounding resources using their mobile devices.

Our work in this area focuses on *user-centric application composition*, a technology that combines services and resources to support users in achieving their everyday activities in ubiquitous environments. A user activity can be as trivial such as watching a movie and chatting in Internet or it can be more complex, such as during a visit to a medical center, where it requires involvement of a composition of multiple services and human resources. This research proposes a system that only requires a user to specify a description of his/her goal following which the system is able to compose an application that helps realizing the goal. Both autonomic and user-controlled composition is supported. The former requires minimum interaction from the user, whereas the latter supposes some degree of user control which will be determined depending on the situation, the user's expertise and his/her needs.

This research facilitates construction of applications with multimodal human-computer interfaces by combining and controlling inputs and outputs from multiple devices. Besides, the research enables applications which rely on ad hoc combinations of services and resources that, for example, were not foreseen at the design time. It also offers higher user comfort and lets the users feel in control by providing them with high-level tools for customizing and composing applications, and mechanisms for controlling these applications at runtime. Over the past year, we focussed on the following areas:

- **Middleware for End-User Composition.** It is important for the end-users to be able to naturally specify their preferences and desired application functionality. This information is essential to define an application structure (i.e., the set of required services and their connections) and formal criteria used for dynamic application composition. In order to address this challenge, we designed the middleware that allows end-users to create simple applications through the discovery of ubiquitous resources and the composition of applications on their handheld devices. The applications are composed in the visual editor, which is based on the jigsaw puzzle metaphor. According to this metaphor, each puzzle piece corresponds to a resource or an action to be performed with a particular resource. Thus, users compose applications by matching and combining various puzzle pieces together. The middleware realizes composed applications at run-time and also provides the basic service discovery and service communication functionality. We reported the design of this middleware and the visual editor in the upcoming book chapter [29].
- **End-User Control Mechanisms.** In order to find a balance between autonomic and user-controlled application composition — the user may need to compose an application manually, restrict the system's autonomy to a certain degree or customize the application further at runtime — we worked on designing user control mechanisms where, the trade-off between user control and system's autonomy is adjusted as required by the user and the situation. We designed four control mechanisms for application composition that combine physical user interfaces (i.e., RFID-based) and interaction techniques on the handheld device. These mechanisms differ from each other on how much the user is willing to be involved in control of application composition so that the user can choose the appropriate mechanism on the account of his/her needs, situation and the application. In order to demonstrate these mechanisms, we designed a prototype and an example multimedia application which were reported in [19].

6.7. Data-driven Macroprogramming for Heterogeneous Sensor Networks

Participants: Animesh Pathak, Iraklis Leontiadis.

Since the goal of WSN macroprogramming research is to make application development easier for the *domain expert*, we believe that it is absolutely necessary to make *easy-to-use toolkits* for macroprogramming available to them in order to both make their task easier, as well as to gain feedback about the macroprogramming paradigms themselves. Although various efforts exist in literature for making WSN application development easier, very few general purpose graphical toolkits for macroprogramming are publicly available for the application developer to choose from. We believe that toolkits supporting alternative paradigms will greatly aid the application developers, who will have a wide-range of programming styles to choose from, depending on application, as well as personal stylistic choice.

As part of our research in the area, we have developed *Srijan* [24], which provides an easy-to-use graphical front-end to the various steps involved in developing an application using the ATaG macroprogramming framework. It consists of the following components:

Task Graph Description GUI: The ability of specifying a WSN application in a graphical manner as interconnected task and data items is a major part of ease-of-use provided by ATaG. In *Srijan*, we have customized the Generic Modeling Environment (GME) so that users can easily developers can easily specify their abstract task graphs, complete with channel and task annotations. Once the details of the task graph are specified, *Srijan* generates an equivalent XML file, which can be used by the other modules.

Network Description, Compilation, and Deployment GUI: The second part of *Srijan* allows the developer to perform a set of actions. First, he can graphically specify the target network description, including the attributes of each node (alternatively, he can upload the specifications in a file). Secondly, the toolkit uses the task graph to generate a separate Java file for each task and data item with auto-generated communication and task-firing code.

Using the GUI, the developer can then invoke the compiler with the necessary parameters (optimization option, randomizer seed, etc.), which results in the generation of a set of Java files, one for each node in the target system, including the task and data code, as well as the customized runtime system modules. Finally, the developer can use *Srijan* to generate the bytecode for each node and deploy it to each Sun SPOT over the air.

We have further added the facility for the toolkit to handle heterogenous nodes [26], and can now support systems consisting of Sun SPOTs as well as regular PC-class nodes, operating on a heterogeneous 2-tier network.

7. Other Grants and Activities

7.1. European Contracts and Grants

7.1.1. FP7 ICT FET CONNECT

Participants: Valérie Issarny, Nikolaos Georgantas, Amel Bennaceur, Animesh Pathak, Rachid Saadi.

- **Name:** CONNECT – *Emergent Connectors for Eternal Software Intensive Networked Systems*
- **URL:** http://www.connect-forever.eu/
- **Related activities:** § 6.5
- **Period:** [February 2009 - July 2012]
- **Partners:** INRIA (UR Rocquencourt, EPI ARLES) – **project coordinator**, Consiglio Nazionale delle Ricerche (Italy), DoCoMo Communications Laboratories Europe Gmbh (Germany), Lancaster University (UK), Thales Communications SA (France), Universita degli Studi L'Aquila (Italy), Technische Universitaet Dortmund (Germany), The Chancellor, Masters and Scholars of the 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 will investigate 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 will be 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. International Research Networks and Work Groups

7.2.1. ASA Associated Team

- **Name:** ASA – *Adaptive SoftwAre*
- **Period:** [created 2007]
- **Participants:** Joint team with Universita dell'Aquila, Dipartimento di Informatica, Italy

The objective of the team is assisting the development of dynamic distributed software systems for next generation ubiquitous communication and computing infrastructures. Software in the near ubiquitous future (Softure) will need to cope with variability, as software systems get deployed on an increasingly large diversity of computing platforms and should further deliver applications ubiquitously. Heterogeneity of the underlying communication and computing infrastructures, mobility and continuously evolving requirements demand new software paradigms that span the entire life-cycle, from development to deployment and execution. Softure must be developed in a way that facilitates both its deployment over heterogeneous networks of heterogeneous nodes, and its interaction with end users, their environment and/or other existing systems, depending on the application domain. Moreover, Softure should be reliable and meet the user's performance requirements and needs. Last but not least, Softure should be dynamic so that the applications they implement can be provisioned ubiquitously, despite the high dynamics of the pervasive networking and computing environment. Looking at the software life cycle, one key issue in this domain appears to be the disappearance of a clear distinction between static and dynamic aspects. Indeed, the adaptability requirement imposed by ubiquity makes software become evolving in nature, therefore introducing a strong interaction between the development environment and the middleware one. The goal of the ASA associated team is to research design and programming techniques and innovative middleware models that can be profitably integrated to support this new generation of software systems.

7.2.2. ForeverSOA Associated Team

- **Name:** ForeverSOA – *A rigorous approach to the evolution of service-oriented software*
- **URL:** http://dmod.cs.uoi.gr/ForeverSOA/index.htm
- **Related activities:** § 6.2
- **Period:** [Created 2009]

- **Participants:** Joint team with University of Ioannina (UoI), Department of Computer Science, Greece

This objective of the team is to study a principled approach for the dynamic maintenance of service-oriented software (i.e., software that is built by composing available services) on the basis of fundamental design principles and middleware that supports their adoption. The need for maintaining service-oriented software may be triggered by changes in the quality requirements of the end-users of service-oriented software (e.g., performance, availability, reliability), or by the the independent evolution of constituent services (e.g., services may be deployed or undeployed at anytime). Specifically, the proposed approach shall consist of:

- A systematic approach that allows to reverse engineer service abstractions out of a given set of available services. A service abstraction shall group concrete services that provide similar functionalities through different interfaces and with different quality characteristics.
- A service discovery middleware that embeds the abovementioned approach and further enables service providers to register available services and service users to discover services exposed as reverse-engineered service abstractions.
- Middleware poly-proxies that serve as local representatives of reverse engineered service abstractions on the side of the service-oriented software that uses them. Poly-proxies shall be dynamically customizable middleware elements that map the changing quality requirements (e.g., performance, reliability, availability) of the end-users of the service-oriented software to the concrete services that can fulfill these requirements. The end-users' quality requirements shall serve as input rules to the dynamically customizable poly-proxies, which will rely on online quality monitoring towards performing the selection of the appropriate concrete services. A second critical property of the poly-proxy middleware elements shall be the ability to dynamically upgrade them with respect to the evolving availability of the concrete services that are registered in / unregistered from the service-discovery mechanism.

7.2.3. ERCIM WG SERENE

- **Name:** ERCIM Working Group – *Software EngineeRing for rEsilieNt systEms*
- **URL:** http://serene.uni.lu/tiki/tiki-index.php
- **Period:** [Created 2004]
- **Participants:** Aabo Akademi (Finland), Birkbeck College (UK), BUTE (Hungary), CNR (Italy), CWI (The Netherlands), FNR (Luxembourg), FORTH (Greece), Fraunhofer FOKUS & IPSI (Germany), INRIA (UR Rocquencourt), LAAS-CNRS (France), National Aerospace University (Ukraine), Nokia Research (Finland), Polit. di Milano (Italy), Poznan University of Technology (Poland), NTNU (Norway), SARIT (Switzerland), SpaRCIM (Spain), SZTAKI (Hungary), University of L'Aquila (Italy), University Mc Gill (Canada), University Mc Master (Canada), University of Florence (Italy), University of Ioannina (Greece), University of Groningen (The Netherlands), University of Newcastle (UK), University Roma Tor Vergata (Italy), VTT (Finland).

SERENE considers resilient systems as open and distributed systems that can dynamically adapt in a predictable way to unexpected events. Engineering such systems is a challenging issue still not solved. Achieving this objective is a very complex task since it implies reasoning explicitly and in a combined way, on system's functional and non-functional characteristics.

SERENE advocates that resilience should be explicitly included into traditional software engineering theories and practices and should become an integral part of all steps of software development. As current software engineering practices tend to capture only normal behavior, assuming that all abnormal situations can be removed during development, new software engineering methods and tools need to be developed to support explicit handling of abnormal situations. Moreover, every phase in the software development process needs to be enriched with phase specific resilience means.

In order not to consider all the scope of software engineering, the SERENE working group focuses on Formal, semi-formal modeling of resilience properties, Frameworks and design patterns for resilience Error handling and fault handling in the software life-cycle, Re-engineering for resilience, Component-based development and resilience, software development processes for resilience, resilience through exception handling in the software life-cycle, Atomic actions, Fault-tolerance, Dynamic Resilience Mechanisms, resilience Prediction, resilience Metadata, Reasoning and adaptation services for improving and ensuring resilience, Intelligent and adaptive approaches to engineering resilient systems, Engineering of self-healing autonomic systems, Dynamic reconfiguration for resilience, Run-time management of resilience requirements, Verification and validation of resilient systems, CASE tools, Model Driven Engineering, and Software architectures for resilience.

7.2.4. ERCIM WG STM

- **Name:** ERCIM Working Group – *Security and Trust Management*
- **URL:** http://www.iit.cnr.it/STM-WG/
- **Period:** [Created 2005]
- **Participants:** British Telecom, CLRC (UK), CNR (Italy), CETIC (Belgium), CWI (The Netherlands), DTU (Denmark), FORTH-ICS (Greece), FNR (Luxembourg), Fraunhofer-SIT (Germany), HP, IBM Research, INRIA (URs Rocquencourt & Sophia Antipolis), IUC (Ireland), L3S (Germany), Marasyk University (Czech Republic), Microsoft EMIC (Germany), NTNU (Norway), Politecnico Torino (Italy), SAP (Germany), SARIT (Switzerland), SICS (Sweden), Siemens Corporate Technology, SparCIM (Spain), SZTAKI (Hungary), VTT (Finland), Eindhoven University of Technology (The Netherlands), University of Milan (Italy), University of Roma Tor Vergata (Italy), University of Trento (Italy), University of Twente (The Netherlands), VCPC, W3C.

The pervasive nature of the emerging Information and Communication Technologies (ICT) expands the well known current security problems on ICT, due to the increased possibilities of exploiting existing vulnerabilities and creating new threats. On the other hand, it poses new problems in terms of possible attack scenarios, threats, menaces and damages. Moreover, the increased virtual and physical mobility of the users enhances their interaction possibilities. Thus, there is a demand for a reliable establishment of trust relationships among the users. Privacy is also a main concern in the current ambient intelligence paradigm: everywhere there are devices interacting with users and information about the users is possibly being gathered by the devices at anytime. All these problems are perceived at different levels of concern by users, technology producers, scientific and governance communities.

This ERCIM Working Group aims at focusing the research of the ERCIM institutions on a series of activities (*e.g.*, projects and workshops) for fostering the European research and development on security, trust and privacy in ICT. These will be among the main issues of current and future research efforts for "security" in a broad sense in Europe (http://www.cordis.lu/security/).

7.3. National Contacts and Grants

7.3.1. SYSTEM@TIC EXOTICUS: Etude et eXpérimentation des Outils & Technologies IMS Compatibles avec les USages

Participants: Valérie Issarny, Mauro Caporuscio, Amel Bennaceur, Pushpendra Singh, Roberto Speicys Cardoso.

- **Name:** EXOTICUS – *Etude et eXpérimentation des Outils & Technologies IMS Compatibles avec les USages*
- **Related activities:** § 6.4
- **Period:** [November 2007 – October 2009]
- **Partners:** Alcatel Lucent - coordinator, Legos, PragmaDev, Archos, Citypassenger, Deveryware, Transatel, ENST, GET/INT, INRIA (CRI Paris-Rocquencourt)

Introduced by the 3GPP standardization forum in relation with 3G, IMS (IP Multimedia Subsystem) is a key element for telecom operators trying to increase the status of their service provider activities, in a highly concurrent marketplace. The incoming IMS should enable the interconnexion of fixed and mobile access networks around a same IP core, and making an open service platform available. The objective of the EXOTICUS project is to tackle the technological locks related to this new architecture, by contributing to:

- bridge the weakness of the norm about processes for the service composition and integration,
- foster the service creation dynamics thanks to new innovative service primitives,
- accelerate the development cycles, experiment new services and assess their use.

This research activities will bring fuel, all along the project, to the experiment platform that is deployed in southern Ile de France, and which is available for both professional (especially from the telecom and automotive sectors) and public population.

7.3.2. ANR SemEUsE: SEMantiquE pour bUS de sErvice

Participants: Nikolaos Georgantas, Nebil Ben Mabrouk, Valérie Issarny, Mauro Caporuscio, Sandrine Beauche.

- **Name:** SEMEUSE – *SEMantiquE pour bUS de sErvice*
- **Related activities:** § 6.3
- **Period:** [December 2007 – May 2010]
- **Partners:** Thales - coordinator, France Télécom R&D, EBM WebSourcing, Université Pierre et Marie Curie - LIP6/MoVe, INSA Lyon, INRIA (CRI Paris-Rocquencourt), ObjectWeb, GET/INT.

The aim of the SemEUsE project is to study a semantic based service infrastructure that will provide the foundational services required for service-oriented applications to exchange information in a ubiquitous, reliable environment. The combination of an emerging semantic service infrastructure and associated engineering techniques will make it possible to produce flexible, mission-critical, software-based service applications that are dependable and manageable, and to provide high levels, business-focused, guaranteed end-to-end quality-of-services for all users.

7.3.3. ANR ITEmIS: IT and Embedded Integrated Systems

Participants: Nikolaos Georgantas, Valérie Issarny, Hamid Ameziani, Mohammad Ashiqur Rahaman.

- **Name:** ITEmIS – *IT and Embedded Integrated Systems*
- **Period:** [January 2009 – December 2011]
- **Partners:** Thales Communications S.A, EBM Websourcing, INRIA ARLES (CRI Paris-Rocquencourt), INRIA ADAM (CRI Lille - Nord Europe), LAAS - CNRS, ScalAgent Distributed Technologies S.A. (SADT), IRIT (IRIT)

Service-Oriented Architecture (SOA), as a key architectural pattern for prompt and rapid integration, is today a cornerstone of the agile Information Technology (IT) wave. Indeed, most of today's greatest successes, in terms of bringing agility to the whole enterprise through its IT backbone, have been provided by SOA and its major technological counterparts that are the Web Services and the Enterprise Service Bus (ESB). At the same time, large control and command systems are envisaged, which may roughly be described as net-centric assemblies of heterogeneous lightweight sensors and actuators along with several large control systems. To accomplish such systems, there is currently a strong need of techniques at the cutting edge of technology that could bring seamless integration and deployment of lightweight embedded applications and IT services in a global agile system of services. In this context, ITEmIS aims at easing the evolution from today's world of separate lightweight embedded applications and IT services to the future world of seamlessly integrated services, thus qualifying and defining a new generation SOA enabling IT and Embedded Integrated Systems (ITEmIS systems). This endeavour is undertaken along three main lines:

1. At business level, where IT/embedded services are integrated into advanced workflows supporting the multi-faceted interoperability and scalability required for ITEmIS systems;
2. At service infrastructure level, by introducing a specialized ESB-based and component-based solution addressing the requirements of the embedded world including deployment; and
3. Transversally for both above levels addressing end-to-end assurance of Quality of Service (QoS) and correctness verification of deployments and workflows at the level of their execution models.

To tackle these three aspects, ITEmIS provides: a service-oriented middleware achieving deployment, run-time integration, and administration of the heterogeneous IT/embedded services into ITEmIS systems; and Model Driven Engineering (MDE)-based meta-models and tools for the modelling, development, deployment, administration and correctness verification of IT/embedded services and ITEmIS systems.

7.3.4. INRIA D2T Action de Developpement Technologique Srijan

Participants: Animesh Pathak, Iraklis Leontiadis.

- **Name:** *Srijan – Data-driven Macroprogramming for Heterogeneous Sensor*
- **Related activities:** § 6.7
- **Period:** [October 2009 – September 2010]
- **Partners:** INRIA (CRI Paris-Rocquencourt, EPI ARLES)

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. In this research, 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 autogenerated 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.

8. Dissemination

8.1. Involvement within the Scientific Community

8.1.1. Program Committees

- Nikolaos Georgantas is PC member of the Middleware ’09, DAIS ’09, ICSOFT ’09 and CFSE-7 international conferences;
- Nikolaos Georgantas is PC member of the RoSOC-M ’09, SIPE’ 09, M-MPAC ’09, MW4SOC ’09 and CAMS ’09 international workshops;
- Valérie Issarny is the PC chair of ESEC/ACM SIGSOFT FSE 2009, the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, August 24-28, 2009, Amsterdam, The Netherlands;
- Valérie Issarny is PC member of the ICSE ’09, ICSOC ’09, IFIPTM ’09 & ’10, Middleware ’09, MP2P ’09, MW4SOC ’09, WICSA ’09, EPEW ’10, FMOODS ’10, SC ’10, Serene ’10, FASE ’11 international conferences;

- Animesh Pathak is PC member of ICPADS ’09, S-Cube ’09, ICISTM ’10, and SECON ’09 & ’10 international conferences

8.1.2. Other Activities

- Nikolaos Georgantas is Demonstrations & Posters co-chair of ESEC/FSE ’09;
- Nikolaos Georgantas is member of the PhD monitoring committee at INRIA Paris-Rocquencourt;
- Nikolaos Georgantas is member of the organizing committee of the monthly colloquium at INRIA Paris-Rocquencourt;
- Nikolaos Georgantas is associate editor of the International Journal of Ambient Computing and Intelligence (IJACI);
- Valérie Issarny is associate editor of ACM Computing Surveys;
- Valérie Issarny is expert for the EC FP6 & FP7 ICT theme.
- Valérie Issarny is member of the Steering Committee of the Middleware and ESEC/FSE conferences
- Valérie Issarny is coordinator of the EC FP7 FET IP CONNECT;
- Valérie Issarny is member of the INRETS scientific council & “Commission d’évaluation des chercheurs”;
- Valérie Issarny is member of the LIESP scientific council;
- Valérie Issary is member of the GDR GPL scientific council ;
- Valérie Issarny is member of the evaluation committee of the ANR ARPEGE call on “Systèmes Embarqués et Grandes Infrastructures” in 2009, and of the evaluation committee of the ANR “Programme Blanc” and “Programme Jeunes Chercheuses et Jeunes Chercheurs” for the 2009 & 2010 call;
- Animesh Pathak was the Production Chair of HiPC 2009 & 2010, and Web Publicity Chair of DCOSS 2009 & 2010;

8.2. Teaching

- Nebil Ben Mabrouk gives a course on Distributed Objects Architectures (laboratory). Final year of the five-year computer engineering degree at the Institut des Sciences et Techniques des Yvelines of the University of Versailles Saint-Quentin en Yvelines;
- Nikolaos Georgantas gives a course on Distributed Objects Architectures (lectures). Final year of the five year computer engineering degree at the Institut des Sciences et Techniques des Yvelines of the University of Versailles Saint-Quentin en Yvelines;
- Valérie Issarny gives a course on Software Architectures for Distributed Systems (lectures), as part of the SAL course of the Master 2 COSY of the University of Versailles Saint-Quentin en Yvelines.

8.3. Internships

During the year 2009, members of the ARLES project-team supervised the work of the following student interns:

- Hamid Ameziani, *Advanced Coordination Models for Information Technology and Embedded Integrated Systems*, Pierre et Marie Curie (UPMC), Paris 6, France.
- Mahanth Gowda, *Supporting Heterogeneity in Data-driven Sensor Network Macroprogramming*, Institute of Technology, Banaras Hindu University, India.
- Elena Kuznetsova, *QoS-aware Service Composition in Pervasive Environments*, KTH Royal Institute of Technology, Sweden.
- Mehrez Lachheb, *Socially-aware Mobile Data Sharing Services*, Ecole Supérieure de Sciences et Techniques de Tunis, Tunis.
- Amir Seyedi, *A People Recommender System for Mobile Social Networks*, Université Paris Sud 11, France.

8.4. Awards

- Pierre-Guillaume Raverdy, formerly ARLES team member, was awarded the Lauréat OSEO-Emergence 2009 award (http://www.inria.fr/actualites/2009/oseo/) relating to technology transfer of ARLES research results in the area of middleware solutions for wirelessly networked handheld devices.

9. Bibliography

Major publications by the team in recent years

[1] S. BEN MOKHTAR, N. GEORGANTAS, V. ISSARNY. *COCOA: COnversation-based Service Composition in PervAsive Computing Environments with QoS Support*, in "Journal of Systems and Software, Special Issue on ICPS'06", vol. 80, n^{o} 12, 2007, p. 1941–1955.

[2] 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", vol. 81, n^{o} 5, 2008, p. 785-808.

[3] Y.-D. BROMBERG, V. ISSARNY. *INDISS: Interoperable Discovery System for Networked Services*, in "Proceedings of ACM/IFIP/USENIX 6th International Middleware Conference (Middleware'2005)", 2005.

[4] M. CAPORUSCIO, P.-G. RAVERDY, H. MOUNGLA, V. ISSARNY. *ubiSOAP: A Service Oriented Middleware for Seamless Networking*, in "Proceedings of 6th International Conference on Service Oriented Computing (ICSOC'08)", 2008.

[5] D. CHARLET, V. ISSARNY, R. CHIBOUT. *Energy-efficient middleware-layer multi-radio networking: An assessment in the area of service discovery*, in "Comput. Netw.", vol. 52, n^{o} 1, 2008, p. 4–24, http://dx.doi.org/10.1016/j.comnet.2007.09.018.

[6] 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.

[7] V. ISSARNY, D. SACCHETTI, F. TARTANOGLU, F. SAILHAN, R. CHIBOUT, N. LEVY, A. TALAMONA. *Developing Ambient Intelligence Systems: A Solution based on Web Services*, in "Automated Software Engg.", vol. 12, n^{o} 1, 2005, p. 101–137, http://dx.doi.org/10.1023/B:AUSE.0000049210.42738.00.

[8] P.-G. RAVERDY, V. ISSARNY, R. CHIBOUT, A. DE LA CHAPELLE. *A Multi-Protocol Approach to Service Discovery and Access in Pervasive Environments*, in "Proceedings of MOBIQUITOUS - The 3rd Annual International Conference on Mobile and Ubiquitous Systems: Networks and Services", 2006.

[9] F. SAILHAN, V. ISSARNY. *Energy-aware Web Caching over Hybrid Networks*, in "Handbook of Mobile Computing", CRC Press, 2004.

[10] F. SAILHAN, V. ISSARNY. *Scalable Service Discovery for MANET*, in "Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications (PerCom'2005)", 2005.

Year Publications

Doctoral Dissertations and Habilitation Theses

[11] M. FREDJ. *Reconfiguration dynamique des architectures orientées services*, Universite Pierre et Marie Curie, December 2009, Ph. D. Thesis.

[12] R. SPEICYS-CARDOSO. *A service-oriented middleware for privacy protection in pervasive computing*, Universite Pierre et Marie Curie, June 2009, Ph. D. Thesis.

Articles in International Peer-Reviewed Journal

[13] A. PATHAK, V. K. PRASANNA. *Energy-Efficient Task Mapping for Data-driven Sensor Network Macroprogramming*, in "IEEE Transactions on Computers", 2010.

[14] A. ZARRAS, P. VASSILIADIS, V. ISSARNY. *Modeling and analyzing reliable service-oriented processes*, in "International Journal on Business Process Integration and Management.", 2009.

International Peer-Reviewed Conference/Proceedings

[15] D. ATHANASOPOULOS, A. ZARRAS, V. ISSARNY. *ForeverSOA: Towards the Maintenance of Service Oriented Software*, in "Proceedings of the 3rd CSMR Workshop on Software Quality and Maintenance (SQM)", 2009.

[16] D. ATHANASOPOULOS, A. ZARRAS, V. ISSARNY. *Service Substitution Revisited*, in "Proceedings of the 24th IEEE/ACM International Conference on Automated Software Engineering (ASE)", 2009.

[17] M. AUTILI, M. CAPORUSCIO, V. ISSARNY. *Architecting Service Oriented Middleware for Pervasive Networking Environments*, in "Proceeding of the 31st International Conference on Software Engineering (ICSE'09), Principles of Engineering Service Oriented Systems (PESOS'09)", 2009.

[18] A. BENNACEUR, P. SINGH, P.-G. RAVERDY, V. ISSARNY. *The iBICOOP middleware: Enablers and Services for Emerging Pervasive Computing Environments*, in "Proceedings of IEEE PerWare'09", 2009.

[19] O. DAVIDYUK, I. SANCHEZ, J. IMANOL, J. RIEKKI. *CADEAU: Collecting and Delivering Multimedia Content in Ubiquitous Environments*, in "Pervasive 2009, the Seventh International Conference on Pervasive Computing (Video), Nara Japon", Springer, 2009, http://hal.inria.fr/inria-00372224/en/.

[20] V. ISSARNY, B. STEFFEN, B. JONSSON, G. BLAIR, P. GRACE, M. KWIATKOWSKA, R. CALINESCU, P. INVERARDI, M. TIVOLI, A. BERTOLINO. *CONNECT Challenges: Towards Emergent Connectors for Eternal Networked Systems*, in "14th IEEE International Conference on Engineering of Complex Computer Systems", 2009.

[21] N. KOKASH, R. S. CARDOSO, V. ISSARNY, P.-G. RAVERDY. *A Flexible QoS-aware Routing Protocol for Infrastructure-less B3G Networks*, in "Proceedings of ACM SAC - The 24th Annual ACM Symposium on Applied Computing - Track on Mobile Computing and Applications", 2009.

[22] N. B. MABROUK, N. GEORGANTAS, V. ISSARNY. *A Semantic End-to-End QoS Model for Dynamic Service Oriented Environments*, in "Principles of Engineering Service Oriented Systems (PESOS'09), held in conjunction with the International Conference on Software Engineering (ICSE'09)", 2009.

[23] N. B. MABROUK, N. GEORGANTAS, V. ISSARNY. *QoS-aware Service Composition in Dynamic Service Oriented Environments*, in "Middleware'09: Proceedings of the International Conference on Middleware", December 2009.

[24] A. PATHAK, M. K. GOWDA. *Srijan: a graphical toolkit for sensor network macroprogramming*, in "ESEC/SIGSOFT FSE (Demo)", 2009, p. 301-302.

[25] R. SPALAZZESE, P. INVERARDI, V. ISSARNY. *Towards a Formalization of Mediating Connectors for on the Fly Interoperability*, in "Joint Working IEEE/IFIP Conference on Software Architecture 2009 & European Conference on Software Architecture 2009, Cambridge United Kingdom", 2009, http://hal.inria.fr/inria-00404308/en/, CONNECT.

Workshops without Proceedings

[26] M. K. GOWDA, A. PATHAK. *Supporting Heterogeneity in Data Driven Sensor Network Macroprogramming*, in "Student Research Symposium at International Conference on High Performance Computing, HiPC 2009 (Poster)", December 2009.

[27] V. ISSARNY, C. CONNECT. *CONNECT: Emergent Connectors for Eternal Software Intensive Networked Systems*, in "FET'09 - The European Future Technologies Conference and Exhibition, Prague Czech Republic", European Commission, 2009, http://hal.inria.fr/inria-00379423/en/, CONNECT.

Scientific Books (or Scientific Book chapters)

[28] S. BEN MOKHTAR, N. GEORGANTAS, V. ISSARNY, P.-G. RAVERDY, M. AUTILI. *Service Discovery in Pervasive Computing Environments*, in "At Your Service: Service-Oriented Computing from an EU Perspective", The MIT Press, Cambridge/Massachusetts, London/England, 2009, ISBN 978-0-262-04253-6.

[29] O. DAVIDYUK, N. GEORGANTAS, V. ISSARNY, J. RIEKKI. *MEDUSA: Middleware for End-User Composition of Ubiquitous Applications*, in "Handbook of Research on Ambient Intelligence and Smart Environments: Trends and Perspectives", F. MASTROGIOVANNI, N. CHONG (editors), IGI Global, 2010, http://hal.inria.fr/inria-00432675/en/.

[30] M. FREDJ, A. ZARRAS, N. GEORGANTAS, V. ISSARNY. *Dynamic Maintenance of Service Orchestrations*, in "SISS 2009 Handbook of Research on Service Intelligence and Service Science: EvolutionaryTechnologies and Challenges.", IGI Global, 2009.

[31] N. GEORGANTAS, V. ISSARNY, J. KANTOROVITCH, J. KALAOJA, I. ROUSSAKI, I. PAPAIOANNOU, D. TSESMETZIS. *Amigo: Interoperable semantic services for the smart home environment*, in "At Your Service: Service-Oriented Computing from an EU Perspective", The MIT Press, Cambridge/Massachusetts, London/England, 2009, ISBN 978-0-262-04253-6.

Books or Proceedings Editing

[32] H. VAN VLIET, V. ISSARNY (editors). *Proceedings of the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering, 2009, Amsterdam, The Netherlands, August 24-28, 2009*, ACM, 2009.