INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team Arles

Software Architectures and Distributed Systems

Rocquencourt

THEME 1A

Activity Report

2003

Table of contents

1. Team

Head of project-team
Valérie Issarny [Research director, INRIA]

Administrative assistant
Sylvie Loubressac [INRIA]

INRIA research scientist
Nikolaos Georgantas [Junior research scientist, since October 2003]

Part-time research scientist (partner)
Nicole Levy [Professor UVSQ (University of Versailles-Saint Quentin en Yvelines)]

Post-doctoral fellow
Nikolaos Georgantas [INRIA/*région IdF* fellowship, until September 2003]

Project technical staff
Rafik Chibout [INRIA-OZONE, since November 2003]
Daniele Sacchetti [INRIA-OZONE]

Ph.D. Students
Raghav Bhaskar [INRIA fellowship, joint PhD with CODES project-team, Ecole Polytechnique]
Malika Boulkenafed [INRIA/*région IdF* fellowship, University of Paris 6]
Yérom-David Bromberg [INRIA fellowship, UVSQ, since November 2003]
Jinshan Liu [INRIA fellowship, UVSQ]
Françoise Sailhan [INRIA/*région IdF* fellowship, University of Paris 6]
Ferda Tartanoglu [INRIA fellowship, University of Paris 6]

2. Overall Objectives

The development of distributed software systems remains a complex task, which is not solely due to the systems' inherent complexity (e.g., heterogeneity, concurrency), but also to the systems' continuous evolution (e.g., integration of new technologies, changing environment in the mobile context). It is thus necessary to offer solutions to the two following issues:

- Supporting the rigorous development of distributed software systems by providing languages for systems modeling together with associated methods and tools for reasoning about the systems' functional and non-functional properties;
- Offering middleware infrastructures for both leveraging the complexity associated with the management of distributed resources and dealing with the efficient integration of new technological development.

The ARLES project-team addresses the two above issues, investigating languages, methods, tools and middleware architectures to assist the development of distributed software systems that are efficient (in terms of both resource usage and delivered quality of service) and dependable. Our approach relies on the development of distributed systems from their architectural description, leading to study the development of distributed systems through composition. This choice is motivated by two factors:

- Our experience in architecture-based development of distributed systems has convinced us about the benefit of the approach regarding the robustness and performance of the resulting systems. Systems' robustness comes from the ability to practically exploit formal methods for modeling the systems' architectures and hence to reason about the systems' behavior. Systems' efficiency results from the possibility to specialize the systems' composition according to both the applications' requirements and the runtime environment, and hence to integrate only necessary functionalities within the system, and further tune their realization according to available resources.

- Practically, the emergence of standard architectures for distributed systems and in particular supporting middleware, leads to the definition of reusable COTS (Commercial Off The Shelf) components for the implementation of both application-related and middleware-related functionalities. In addition, a number of systems are built out of the integration of legacy systems, as in particular witnessed in the context of information systems. The development of distributed systems thus becomes oriented towards the composition of system components and/or running system instances.

The research activities of the ARLES project-team are more specifically centered around the development of distributed systems enabling the ambient intelligence vision. Ambient intelligence is an emerging user-centric service provision paradigm that aims at enhancing the quality of life by seamlessly offering relevant information and services to the individual, anywhere and at anytime. Systemically, this is realized as a synergistic combination of intelligent-aware interfaces, ubiquitous computing and ubiquitous networking. The intelligent-aware property applied to interfaces enables: (i) support of natural ways of interaction, e.g., through speech and gesture; (ii) automatic adaptation to user's personal preferences; and (iii) proactiveness, stimulated by the presence of people, their location and their activities, instead of simple reactiveness to conventional ways of interaction, such as a keystroke or a mouse click. The ubiquitous (alternatively called pervasive) property applied to both computing and networking implies a useful, pleasant and unobtrusive presence of the system everywhere – at home, en route, in public spaces, in the car, at work, and wherever else the electronic environment support extends. The computing and networking facilities are distributed and accessible in wide varieties, as needed. The ubiquitous computing and networking model incorporates the mature paradigms of mobile and nomadic computing, and distributed systems.

While a number of base enablers such as wearable and handheld computers, wireless communication, sensing mechanisms are already commercially available for deploying base infrastructures supporting the ambient intelligence vision, the development of ambient intelligence software systems still raises numerous scientific and technical challenges due to the specifics of ambient intelligence. In addition to traditional requirements for the software systems like dependability, the software systems shall deal with: mobility of users, increasing heterogeneity in devices, networks and software infrastructures, varying user and application requirements, diverse contexts of service provision, and natural interaction integrating multi-modal interfaces and exploiting knowledge about the user and his/her environment. The above requirements reveal the highly dynamic character of ambient intelligence systems, which shall be accommodated by the overall software system architecture. Specifically, ambient intelligence software systems must comprehensively offer the following features: be self-adaptive according to a combined user-centric and computer-centric context so that service delivery continuously adapts to the highly changing situation of users, be dependable, and provide multi-modal interfaces for natural interaction with users. Developing systems with such features has given rise to extensive research since the end of the nineties, following the concern of seamlessly and effectively combining the numerous technologies that emerged for the benefit of users, as opposed to putting increased burden on them for mastering the increasing complexity of technologies. This concern is the key point of the ambient intelligence vision, as well as the ones of pervasive and autonomic computing. Despite the large interest of the research community in addressing the challenges raised by these visions since their emergence, the open issues that arose at that time have yet to be addressed. Regarding specifically the development of software systems for ambient intelligence, a key approach lies in the dynamic composition of the software systems according to the environment. The objective of our work is thus to offer comprehensive solutions to the dynamic composition of distributed systems, decomposing into:

- The definition of software architecture styles dedicated to mobile distributed systems that are dynamically composed. This shall serve eliciting key interaction paradigms and core properties of the architectural elements, further leading to devise associated architecture modeling and supporting middleware infrastructure.
- The definition of an architecture-based development environment offering an Architecture Description Language (ADL) and associated methods and tools for assisting the development of systems

that are dynamically composed, and may in particular be deployed over resource-constrained mobile devices. Results shall ease the modeling of mobile systems and reasoning about the behavior of the systems regarding both functional and non-functional properties. They shall further serve dynamically composing systems according to the environment (e.g., available resources, connectivity).

- The definition of a middleware infrastructure aimed at mobile distributed systems, which integrate mobile nodes that are possibly resource-constrained and communicate via a variety of wireless networks (i.e., infrastructure-based and ad hoc). We are in particular interested in the exploitation of (multi-hop) ad hoc networks, which do not require any infrastructure for accessing remote resources.

3. Scientific Foundations

3.1. Introduction

Key words: *ad hoc networks*, *ambient intelligence*, *dependability*, *distributed systems*, *middleware*, *mobile computing*, *software architecture*, *software engineering*, *system composition*, *Web services*, *wireless networks*.

Research undertaken within the ARLES project-team aims at offering comprehensive solutions to support the development of mobile distributed systems that are dynamically composed according to the environment. This leads us to investigate dedicated software architecture styles from which to derive:

- Architecture description languages for modeling mobile distributed software systems, together with associated methods and tools for reasoning about the systems' behavior and automating the systems' composition, and
- Middleware infrastructures for leveraging 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, and in particular ad hoc networks, on the 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 [7]. 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 that interaction. 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 90s [6]. 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 deriving system implementation and deployment, provided that 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 CASE tools, which enable 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 base ground towards assisting the development of robust distributed systems, which is further eased by middleware infrastructures.

3.2.1. Middleware-based and service-oriented software architectures

The main constituents of any distributed system are the system components offering services to other components requesting service execution. The system's component model then defines the way services are specified and accessed, as well as the way components are identified. Considering distributed systems that are developed nowadays, these functionalities rely on some distributed middleware defining the system model. Available middleware can be classified into three main categories [2] : 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 originally 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 and connectors abstract at least the interaction protocols offered by the middleware). 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, are promising new levels of software integration and interoperability. 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 have already emerged from the W3C[1], which define a core middleware for Web services, partly building upon results from object-based and component-based middleware technologies. These standards relate to the specification of Web services and a supporting interaction protocol. SOAP (Simple Object Access Protocol) defines a lightweight protocol for information exchange that sets the rules of how to encode data in XML, as well as the SOAP mapping to an Internet transport protocol (e.g., HTTP). The specification of Web service interfaces relies on the WSDL (Web Services Description Language) declarative language, which is used to specify: (i) the service's abstract interface that describes the messages exchanged with the service, and (ii) concrete binding information that contains specific protocol-dependent details including the network end-point address of the service. Complementary to the above core middleware for the integration of Web services is UDDI (Universal Description, Discovery and Integration); this specifies a registry for dynamically locating and advertising Web services. Composing Web services relates to dealing with the assembly of existing services, so as to deliver a new service out of them, 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 actually 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, extendibility, 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 due to the concern for supporting business processes, combined with the fact that the composition process deals with the assembly of loosely-coupled autonomous components.

Although Web services have been primarily designed for realizing complex business processes over the Internet, they appear as a promising architectural choice for ubiquitous computing. The pervasiveness of

[1] http://www.w3.org/2002/ws/arch

the Web allows anticipating the availability of Web services in most environments, considering further that they may be hosted on mobile nodes. Hence, this serves as a sound base ground towards dealing with the dynamic composition of services in the mobile 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 infrastructure, 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. Hence, the development of such systems can be assisted with an architecture-based development process in a straightforward way. This is already supported by a number of ADL-based development environments targeting system construction, such as the Aster environment that was developed by members of the ARLES project-team[2]. 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 promoted by middleware infrastructures, is that of assembling components that rely on distinct interaction patterns. This aspect is known as architectural mismatch [3] and is one of the criteria substantiating the need for connectors as first-class entities in architecture description. The abstract specification of connector behavior, as, for instance, supported by the Wright ADL [1], 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 provided with means for solving architectural mismatches, which further promotes software reuse.

Connectors that are implemented using middleware infrastructures 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 [4]. In the same way, this a priori knowledge about middleware architectures enables dealing with the safe dynamic evolution of the middleware architectures according to environmental changes, by exploiting both the support for adaptation offered by novel middleware infrastructures (e.g., reflexive 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 aiding in the system's design and, further, in the software system's assessment and construction. Ongoing work focuses on closer coupling with solutions that are used in practice for the development of software systems. This includes integration of ADLs with the now widely accepted UML standard for system modeling. Still in this direction, coupling with OMG's model-driven architecture should be 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 infrastructures. However, addressing all the features enabled by middleware within the architecture

[2] http://www-rocq.inria.fr/arles/work/aster.html

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 serve dealing with the system evolution at runtime and further assessing the behavior of the resulting system.

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 computing have now been investigated for more than a decade following the emergence of wireless networks and laptops, these have mostly concentrated on adapting existing distributed systems architectures, so that the systems tolerate the occurrence of disconnection. Basically, this had led to devise replication strategies for 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 infrastructures for leveraging the complexity associated with the management of dynamic networks. In this context, ad hoc networking is amongst the most challenging network infrastructures 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 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) 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 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 (i.e., mobile terminals that are accessible in a fixed number of hops). 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.

3.3.2. Managing the network's dynamics

Trends in mobile computing have created new requirements for automatic configuration and reconfiguration of network devices and services. This has led to propose a variety of protocols for lookup and discovery of network 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.

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 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 deploying 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 adapted 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 flood 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 leads to replicate 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 are in particular suited for ad hoc networks remains an open issue. In addition, 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. In this context, we have in particular initiated work on group management over ad hoc networks, which allows to abstractly characterize the mobile network on top of which the application is intended to execute and to manage the network on behalf of the application. 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.

3.3.3. Enforcing dependability

Dependability of systems is defined as the reliance that can justifiably be placed on the service that the system delivers [5], further decomposing 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 make necessary 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 devise 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 (e.g., see [8]). 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 has either 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 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 conveniently be 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 willing to gain but not provide resource access.

4. Application Domains

Key words: *Ambient intelligence*, *distributed systems*, *information systems*, *mobile systems*, *Web services*.

The ARLES project-team targets development support for applications relevant to the ambient intelligence domain, with a special focus on consumer-oriented applications. Architecture-based development of systems of systems is further directly relevant to enterprise information systems, whose composition is mainly static and must deal with 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 in particular solutions for the mobile context.

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 OZONE project (§ 8.1.2), which relate to the extended home environment [25]. We specifically contribute to the development of a demonstrator focused on the away situation, in collaboration with the INRIA IMARA, LED, MAIA, PAROLE and SARDES project-teams. The demonstrator allows for seamless access to, possibly composite, Web services, both in the local and the wide area, from various mobile terminals (e.g., wireless PDAs, terminals in the car) through a multimodal interface combining speech and gesture.

5. Software

5.1. Introduction

For the sake of validation of our research results, our research activities encompass development of related prototypes. Available prototypes relate to: (i) an architecture-based development environment that integrates our previous research results on architecture-based development of information systems over fixed networks (§ 5.2), (ii) a distributed file system that integrates our research results on middleware services for mobile data sharing among peer nodes in the communication range of each other via the underlying WLAN (§ 5.3), and (iii) a middleware infrastructure based on the Web services architecture that integrates our research results on the dynamic composition of mobile distributed systems that offer quality of service properties to users (§ 5.4).

5.2. Architecture-based Development Environment

Participant: Valérie Issarny.

The quality of Information Systems (IS) is characterized by a number of non-functional properties (e.g., performance, reliability, availability, etc.). Assessing the IS quality against these properties imposes the application of quality analysis and evaluation. Quality analysis consists of checking, analytically solving, or simulating models of the system, which are specified using formalisms like CSP, CCS, Markov-chains, Petri-nets, Queuing-nets, etc. However, developers are usually not keen on using such formalisms for modeling and evaluating IS quality. On the other hand, they are familiar with using architecture description languages and object-oriented notations for building IS models. Based on the previous and to render the use of traditional quality analysis techniques more tractable, we have designed, and implemented a prototype of, an architecture-based environment that facilitates the specification and quality analysis of IS at the architectural level [15]. Our environment allows for the description of system architectures using UML, including the specification of quality-related attributes. Formal quality models are then automatically generated from the systems' architectural descriptions, enabling automated analysis of the systems' quality using available tools. The prototype implementation of our environment builds on the Rationale Rose tool, which it extends with: (i) an add-in for the description of systems' software architectures according to stereotyped UML elements aimed at architecture modeling, and (ii) tools for the automated generation of quality models supporting reliability analysis using the SURE/ASSIST tool from NASA, performance analysis using the QNAP tool from SIMULOG, and model checking using SPIN. Prototype implementation of our environment was terminated in 2002, and now serves as a base development environment within the project-team.

Note that the above software development was undertaken as part of our research on architecture-based development of closed distributed systems, and related analysis support at design time. Our research now concentrates on architecture-based development of dynamic, mobile distributed systems, further leading to investigate on-line analysis of systems that are dynamically composed according to the environment and in particular network connectivity.

5.3. AdHocFS: A Distributed File System for Ad Hoc Networks

Participants: Valérie Issarny, Malika Boulkenafed.

The AdHocFS distributed file system has been designed so as to allow exploiting the capabilities of today's wireless networks and in particular the ad hoc mode of WLANs (more specifically, one hop ad hoc networks). As such, AdHocFS supports synchronous collaborative work among mobile nodes that are in the communication range of each other without requiring the presence of any infrastructure. In AdHocFS, the file systems of mobile terminals act as local caches, and mobile terminals that are granted access to common files and are able to communicate through the wireless LAN, cooperate to form an ad hoc distributed file system. Core components of AdHocFS comprise [17]:

- A naming service, which resolves file names into the various locations from which files may be retrieved (i.e., at least the address of the file's home server, local copy if cached, peer terminals in the communication range that store a file copy).
- A group management service, which enables setting up ad hoc groups of trusted mobile terminals that are connected using the wireless LAN in the ad hoc mode.
- A coherency management service, which reconciles copies cached on mobile terminals that belong to the same ad hoc group and enables synchronous, collaborative file sharing among peer nodes of the group.
- A replication service, which manages data availability within an ad hoc group through the creation of preventive replicas according to the profiles of peer nodes, so as to in particular prevent the loss of the files that are being (or will likely be) accessed within the group.

The prototype of AdHocFS is implemented in Objective Caml 3, and builds upon the Extended 2 FS (Ext2) local file system and the OpenSLP implementation of SLP.

AdHocFS resembles and benefits from past work on optimistic replication in mobile systems, since we have adopted a log-based solution for update propagation, which has been proven successful in the area of data management for mobile computing. However, AdHocFS differs from early work in the area by accounting for the specifics of today's WLANs. In particular, this has led us to use a conservative replication scheme within ad hoc groups, as it is more suited to collaborative applications, and allows reducing the communication and energy costs associated with coherency management.

5.4. WSAMI: A Middleware based on Web Services for Mobile Systems

Participants: Valérie Issarny, Rafik Chibout, Daniele Sacchetti.

Enabling the ambient intelligence vision means that consumers will be provided with universal and immediate access to available content and services, together with ways of effectively exploiting them. Concentrating on the software system development aspect, this means that the actual implementation of any ambient intelligence application requested by a user can only be resolved at runtime according to the user's specific situation. Towards that goal, we have introduced the WSAMI middleware [11], which supports the abstract specification of Ambient Intelligence applications in the form of software architectures, together with their dynamic composition according to the environment. The proposed middleware builds on the Web services architecture, whose pervasiveness enables service availability in most environments. In addition, dynamic composition of applications is dealt with in a way that enforces quality of service for deployed applications in terms of security and performance through the systematic customization of connectors that dynamically integrates relevant middleware-related services.

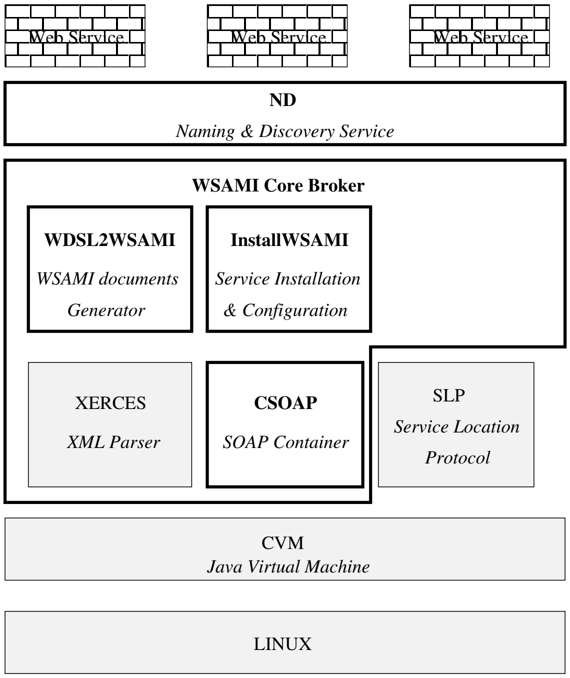

Figure 1. WSAMI core middleware prototype

We have developed a first Java-based prototype of the WSAMI core middleware. We use IEEE 802.11b as the underlying WLAN. The WSAMI core middleware prototype subdivides into: (i) the WSAMI SOAP-based core broker, including the CSOAP SOAP container for wireless, resource-constrained devices, and (ii) the Naming&Discovery (ND) service, including support for connector customization, so as to enforce quality of service through the dynamic integration of middleware-related services over the network's path. Figure 1 depicts the main components of the WSAMI prototype implementation, on top of which Web services execute; grayed components denote available implementations that we reuse, while the components that we have developed are highlighted in bold face. Note that the components developed as part of the WSAMI core broker exist in any Web services platform; a new implementation has been elaborated, so as to allow for execution on resource-constrained devices. The CSOAP SOAP container prototype has been in particular developed to cope with the limitations imposed by CVM[3] and resource-constrained devices. The CSOAP prototype implementation is mainly based on Sun's JAX-RPC specification[4], which aims at enabling the development of SOAP-based interoperable and portable Web services. Together with the CSOAP container, we have developed the InstallWSAMI tool, which provides deployment and configuration functionalities. The utility for deploying and undeploying Web services is based on a simple XML-based language, specifically defined for the purpose of minimizing the complexity of the system. A related tool for automated stub and skeleton files generation is provided to allow an easy development of Web Services on the server side and of the corresponding client applications. The memory footprint of our CSOAP implementation is of 90KB, as opposed to the 1100KB of the Sun's reference implementation. The overall memory footprint of our Web services platform is of 3.9MB, dividing into 3MB for the CVM and 815KB for the Xerces XML parser, in addition to the CSOAP implementation. We have further carried out a number of experiments to investigate the performance of our lightweight Web services platform, and in particular assess it against existing Web services platforms, but also against traditional middleware platforms being considered for mobile environments. First

[3] http://java.sun.com/products/cdc
[4] http://java.sun.com/xml/jaxrpc/

experiments with our prototype are encouraging: (i) the development of ambient intelligence systems using our environment does not add any complexity to the one of Web services, and (ii) performance of the resulting systems is comparable to the one obtained with traditional middleware, and allows for execution on wireless, resource-constrained devices. In addition, the overhead related to the functions handling user mobility (i.e., dynamic service composition relying on service discovery and connector customization for enforcing quality of service) compares to the cost of base Web services access.

Our prototype is being used for the implementation of demonstrator applications in the field of ambient intelligence. It will further be extended with a number of value-added middleware services for mobility management that integrate our research results in the area.

6. New Results

6.1. Introduction

The ARLES project-team investigates solutions in the forms of languages, methods, tools and supporting middleware, to assist the development of distributed systems, with a special emphasis on mobile distributed systems enabling the ambient intelligence vision. Towards that goal, we undertake an approach that is based on the architectural description of software systems, further allowing to deal with the dynamic composition of systems according to the environment. Our research activities thus subdivide into two core activities:

(*i*) Software architectures for distributed systems, where we investigate architectural styles dedicated to mobile distributed systems from which to derive languages for system modeling and related methods and tools for supporting system development (§ 6.2).

(*ii*) Middleware architectures for mobile systems, building upon architectural styles elicited for mobile distributed software systems and further investigating solutions that meet constraints associated with today's wireless networks and devices for their effective exploitation (§ 6.3).

6.2. Software Architectures for Distributed Systems

Participants: Yerom-David Bromberg, Nikolaos Georgantas, Valérie Issarny, Nicole Levy, Jinshan Liu, Ferda Tartanoglu.

Building upon our past work on modeling software architectures of closed distributed systems for supporting the systems' analysis and synthesis, we are investigating architectural styles of dependable, mobile, distributed systems that can be dynamically composed out of resources available in the network. Our work in this area over year 2003 has focused on two complementary issues: (i) supporting the dependable composition of systems with a special focus on composite Web services (§ 6.2.1), and (ii) modeling networked services so as to support QoS-aware service discovery and composition in ad hoc networks (§ 6.2.2).

6.2.1. Dependable composition of Web services

Since the emergence of Web services, we have been investigating support for the development of dependable composite Web services, as they will undoubtedly become a major class of open distributed systems. This will further serve as a base ground towards solutions assisting the development of dependable, mobile composite systems.

Our solution to the dependable composition of Web services is based on forward error recovery [14]. While exploiting possible support for fault tolerance of composed services (e.g., transactional support at the level of each service), the proposed approach has no impact on the autonomy of the individual Web services. Our solution primarily lies in system structuring in terms of co-operative atomic actions, which have a well-defined behavior, both in the absence and in the presence of service failures. More specifically, we define the notion of Web Service Composition Action (WSCA), based on the Coordinated Atomic Action notion, which allows structuring composite Web services in terms of dependable actions [20]. Fault tolerance can then be obtained as an emergent property of the aggregation of several, potentially non-dependable, services. We have further introduced a framework enabling the development of composite Web services based on WSCAs, consisting of the XML-based WSCAL language for the specification of WSCAs and related tools for implementation and deployment of composite services [21].

The WSCAL language currently allows structuring dependable composite services and specifying forward error recovery actions under the occurrence of faults. It is then up to the developer to specify the recovery actions according to the specifics of the application. Further guidelines are offered for this activity in the light of work in the area of multi-database systems. In addition, we have conducted a preliminary study towards offering automated support for undoing the effect of compensable operations over shared Web services [29].

6.2.2. Dynamic composition of mobile systems

The advent of light-weight terminals (e.g., PDA) with integrated communication capabilities facilitates service access and hosting anytime, anywhere. However, effective service access requires dealing with the service's Quality of Service (QoS), including related resource consumption. To support such a facility, we have introduced a comprehensive framework for QoS-aware service location in mobile ad hoc networks (MANET) [18]. Our framework divides into: (i) QoS specification that is expressive enough to capture most significant QoS-related properties, and concise enough to have minimal computation cost associated with QoS management, and (ii) a benefit function for evaluating the overall benefit of service instances available in the network from the standpoint of QoS regarding both the user's perspective and resource consumption on hosts. We have further studied incentive schemes for stimulating service provision in MANET. Mobile distributed systems that provide access to information and services spread among autonomous devices is of paramount importance to the operation of MANET. Unfortunately, cooperative behavior required by the above system implies resource consumption (e.g., battery), which is not in the interest of the autonomous devices. Mechanism design, which studies how to design systems so that agents' selfish behavior results in desired system-wide social goals, is a suitable approach for our purpose. In that context, we are investigating how to apply Vickrey auction, a well-known incentive-compatible mechanism design for service allocation in MANET. Our choice of Vickrey auction is justified because of its valuable theoretical properties, e.g., it is always best for the agents to reveal the true valuation. However, application of Vickrey auction on MANET is challenged by the inherent shortcomings of Vickrey auction and characteristics of MANET. We are thus exploring solutions to these challenges so as to design an auction system for service allocation in MANET.

6.3. Middleware Architectures for Mobile Systems

Participants: Raghav Bhaskar, Malika Boulkenafed, Rafik Chibout, Nikolaos Georgantas, Valérie Issarny, Jinshan Liu, Daniele Sacchetti, Françoise Sailhan.

In order to ease the development of mobile systems enabling the ambient intelligence vision, we are investigating supporting core middleware infrastructure and associated services. Our work in this area over year 2003 has concentrated on the following aspects: (i) development of a core middleware infrastructure based on the Web services architecture that enables the situation-sensitive composition of (possibly mobile) networked services (§ 6.3.1), (ii) elicitation of middleware services for mobile data sharing that increase data availability while minimizing resource consumption (§ 6.3.2), (iii) design of a service location protocol for ad hoc networks (§ 6.3.3), and (iv) design of a group management service for ad hoc networks that is generic with respect to membership constraints and manages the network's dynamics on behalf of applications (§ 6.3.4).

6.3.1. Middleware infrastructure for ambient intelligence systems

Enabling the ambient intelligence vision means that consumers will be provided with universal and immediate access to available content and services, together with ways of effectively exploiting them. Concentrating on the software system development aspect, this means that the actual implementation of any ambient intelligence application requested by a user can only be resolved at runtime according to the user's specific situation. To support such a feature, we have introduced a base declarative language and associated core middleware, which respectively allow for the abstract specification of ambient intelligence applications, and for the dynamic composition of applications according to the environment. Our solution primarily builds on results of component-based software engineering and architecture-based development of software systems, which have been proven successful for the development of distributed software systems: ambient intelligence applications are developed in terms of composition of services that are defined through their abstract interfaces. The ambient intelligence requirement of enabling any-time, any-where access to applications from any terminal further leads to bind with related services instances at runtime, according to the environment (including network connectivity) in which services are requested. Such a facility then requires a software technology that is pervasive enough for being able to rely on both consistent specification and availability of services in most environments, so as to actually support any-time, any-where discovery of service instances from abstract interfaces. This has led us to base our solution on the Web, and more specifically on the Web services architecture. Our main design objective was then to offer a solution that could be deployed in any environment, and effectively supported by mobile, resource-constrained devices.

We have introduced the XML-based WSAMI declarative language for the specification of Web services taking part in the realization of ambient intelligence applications, together with associated core SOAP-based WSAMI middleware [11]. The language allows for dynamically retrieving instances of services matching a requested application by comparing only URIs of XML documents. Hence, the cost associated with the dynamic composition of applications, in terms of resource consumption (and in particular energy), is kept very low, which is mandatory for wireless devices. The language further allows for composing applications that guarantee security and performance properties, which we consider as prime requirements for the actual acceptance of ambient intelligence by consumers. Actual composition of applications at runtime relies on the core middleware, which amounts to supporting SOAP and to providing a naming&discovery service for the dynamic retrieval of service instances, both in the local area over WLAN and in the wide area, according to the network connectivity and associated cost. We have implemented a first prototype of the WSAMI middleware (see § 5.4) so as to practically validate our solution with respect to offered development support and performance.

Supporting the development of ambient intelligence or pervasive computing systems has given rise to extensive research over the last couple of years, which has led to introduce a number of complex middleware-related services that place high demand on the underlying platform and hence limit deployment in most environments. Our contribution lies in the definition of a minimal middleware infrastructure for the actual dynamic composition of services, i.e., a naming&discovery service in addition to SOAP, which allows for its wide deployment but also incurs minimal overhead in terms of resource consumption and response time. Additional middleware-related services may be exploited for increased quality of service, depending on the specific target application, but they do not have to be supported in all environments. Our solution resembles work in the area of service discovery, given the base support offered by the platform. However, results in the area target local area networks, while we target composition of services that may be retrieved both in the local and the wide area. In addition, by building upon the Web services architecture, availability of services is promoted. We are currently working on complementing our core solution with support for off-line analysis of Web services so as to enforce robustness of the composed applications, regarding in particular behavioral matching with respect to both functional and non-functional properties. We are also investigating the exploitation of user and service profiles for the naming&discovery service, which will in particular enable an enhanced service selection process with respect to matching user and service requirements, and the integration of advanced prefetching techniques for enhancing response time.

6.3.2. *Mobile data sharing*

Ad hoc networks enable users equipped with lightweight computing devices and wireless interfaces to form a temporary network without the need for any established infrastructure. It is then quite interesting to exploit such flexibility to allow users to share and manipulate data in a collaborative manner (e.g., working meeting, network gaming, etc.), provided that some guarantees regarding security are given. Towards that goal, we have designed a set of supporting core middleware services, further elaborating their integration into the prototype implementation of a mobile, distributed file system aimed at ad hoc networks (see § 5.3) [9]. Core services for collaborative data sharing over ad hoc networks subdivide into [16]:

- **Management of secure, mobile ad hoc groups:** The secure group management functionality includes discovery of peer mobile terminals that are in the communication range of each other, using some existing service discovery protocol. Groups are restricted to one hop ad hoc networks, since we consider that the collaborating peers are usually located in the local communication range of each other. However, every peer is free to leave the group, as well as, new trusted peers can join it at any time. To form a collaborative group, all the peer terminals should authenticate themselves through digital certificates. Then, peers that can trust each other to build a secure ad hoc group in order to share data and collaborate, enter into a Group Key Agreement (GKA) protocol to come up with a common secret. Group management is periodic, so as to deal with the dynamics of the network.
- **Replication and coherency management:** Data sharing within the ad hoc group is carried out by making sure that each peer within the group has complete knowledge of all the data cached/stored within the group. Access to such data from any of the mobile terminals belonging to the group, leads to copying it locally, if not already cached. Data coherency is maintained by enforcing a conservative coherency protocol, which takes into account mobile devices and wireless network constraints, and provides an effective support for collaborative work, since collaborating users must have the same version of the shared data. However, local data can be manipulated (read/write) independently within disjoint groups, provided that data are synchronized when groups (including singleton groups) merge. Update propagation for given shared data occurs only when any peer member of the group tries to access the data, because if a group member updates its local copy of particular data, propagation of this update is not necessary if none of the other group members wants to read or update the same data (either cached locally or not). Furthermore, updates are propagated only to the peer which is to access the data. This follows from our aim to save energy by reducing communication among peers.
- **Data availability management:** To enhance data availability, an adaptive data replication protocol with respect to mobile devices constraints is used. In the context of collaborative work within mobile ad hoc groups, data may become unreachable if the peers storing it suddenly disconnect. On the other hand, excessive or systematic data replication in order to address unforeseen disconnection leads to unnecessarily overloading the group's peers, and, in particular, to greatly increasing their energy and storage space consumption. Thus, useful data for collaborative work within a group has to be rationally replicated on peer devices, with respect to the devices' resources. This is addressed through the management of peers' profiles, which serve to identify whether peers can be involved in increasing data availability, according to their specific situation. Preventive replication of a shared file is then undertaken, if the peer holding the file is to leave the group, on the peer that is the most suited regarding availability of local resources, mobility, and probability of locally accessing the file.

6.3.3. Resource discovery over ad hoc networks

Ad hoc networks are well suited to support ambient intelligence applications, that is, to provide an immediate access to resources (i.e., content or services) anywhere, at anytime, at low cost. In this context, automatic discovery of resources within the network plays an essential role. Conventional resource discovery solutions are not well suited for ad hoc networks, as they, in particular, use broadcasting to discover service providers. This results in the unavailability of the ad hoc network, which is induced by broadcast storms. Our work focuses on designing a resource discovery system for (multi-hop) ad hoc networks. Our solution is designed so as to limit the induced traffic load, particularly when the number of users increases dramatically. Our solution is based on a subset of self-organized devices (called directories), which are periodically elected to store information about networked resources for the surrounding devices. Then, devices can access information from directories without flooding the network to discover resources. In addition, the system is designed to cope with moving resources, and, thus, related mobility-induced failures, by accounting for the existence of resource replicas within the network together with the quality of service offered by eligible resources.

Another critical issue in ad hoc networks is to enable users to easily access information from both the local and the wide area (e.g., the Internet). However, we have not yet reached the point where anywhere, anytime network access is actually offered. Infrastructure-based wireless networks use fixed network access points with which mobile terminals interact for communicating. Unfortunately, the unavailability of a base station results in network failure. Ad hoc networking may then be exploited for accessing resources available in the local area, which comes at no cost for users, and possibly accessing a WLAN base station to reach resources available in the wide area. The issue that we are addressing is on setting up an ad hoc network of mobile terminals that cooperate to access resources from the local network, and also to offer utilities intended to discover resources in the global network (i.e., the Internet), when needed. This requires interaction with the base station to gain access to the rich set of available Internet resources, when sought resources are not available in the local area. In this context, we have first concentrated on how to improve the Web latency using a WLAN, exploiting both the ad hoc and infrastructure-based capabilities of the network [19]. Our main design objective was to minimize the energy cost of peer-to-peer communication among mobile terminals, so as to allow for inexpensive access when a fixed access point is not available in the communication range of the mobile terminal [13].

6.3.4. Group management

Mobile ad hoc networks (MANET) pave the way for pervasive computing due to their inherent support for any-time, any-where network access for mobile users. Nonetheless, the highly dynamic nature of mobile ad hoc networks poses tremendous challenges for the development of applications, since the application's context keeps changing over time. One approach to master this complexity lies in the management of groups over MANET, i.e., applications execute on top of groups that manage the dynamic execution context, including mobility-related failures. There has been extensive research on group management and related group communication services in the context of fixed networks, with special emphasis on providing availability properties. However, proposed solutions cannot be applied directly to mobile wireless networks due to the network's highly dynamic topology. This has led to adapting the management of group membership to the specifics of MANET. Various solutions towards group management over MANET have, then, been investigated over the last couple of years, each targeting specific applications. However, a distinctive set of key attributes may be identified for MANET-based groups, which may further be exploited to design a generic group service that is to be customized by applications [23].

We have elicited key attributes for group management over MANET, in particular based on applications published in the literature. Those attributes amount to setting membership constraints in relation with the location, connectivity, authentication and supported QoS of group members. We have then introduced a group service that is generic with respect to membership constraints, and realizes three basic functions: discovery of group members, initialization of the group, and management of the group's dynamics. Implementation of the generic group service has further been addressed in the context of the WSAMI middleware (§ 6.3.1), which is aimed at mobile distributed computing and is based on the Web services architecture. Finally, we have studied two instances of groups that build on our generic group service and allow supporting ambient intelligence scenarios that are respectively related to mobile collaborative work and QoS management in the home network.

Group management further requires adequate security support. In this context, we have undertaken a study on security mechanisms for MANET in collaboration with the CODES project-team. First results relate to the design of a group key agreement protocol among members of the group, which accounts for resource constraints of participating nodes [22].

8. Other Grants and Activities

8.1. European Initiatives

8.1.1. IST DSoS

Participants: Nikolaos Georgantas, Valérie Issarny, Nicole Levy, Ferda Tartanoglu.

- **Name:** IST DSoS – Dependable Systems of Systems
- **URL:** http://www.newcastle.research.ec.org/dsos/index.html
- **Related activities:** §6.2.1, §5.2
- **Period:** [April 2000 - May 2003]
- **Partners:** University of Newcastle upon Tyne (UK) – project coordinator, CNRS-LAAS (France), INRIA (UR Rocquencourt), LRI (France), QinetiQ (UK), Technical University of Vienna (Austria), University of Ulm (Germany).

The overall objective of the DSoS project was to develop significantly improved means for composing a dependable "system of systems" (SoS) from a set of largely autonomous component computer systems. A system of systems provides new emerging services to its users, in addition to the services provided by its component systems. This project aimed to ensure that both types of services are provided with a level of dependability matching specified user requirements. Dependability in this context encompasses reliability, security and maintainability, though the project focused mainly on the first of these system characteristics, with some emphasis on timeliness as well as functionality issues.

8.1.2. IST OZONE

Participants: Rafik Chibout, Nikolaos Georgantas, Valérie Issarny, Daniele Sacchetti.

- **Name:** IST OZONE – *New technologies and services for emerging nomadic societies*
- **URL:** http://www.extra.research.philips.com/euprojects/ozone/
- **Related activities:** §6.3.1, §5.4
- **Period:** [November 2001 - August 2004]
- **Partners:** Philips Research Eidhoven (The Netherlands) – project coordinator, Epictoid (The Netherlands), IMEC (Belgium), INRIA (URs Loraine, Rennes, Rhône-Alpes, Rocquencourt), Technical University of Eindhoven (The Netherlands), THOMSON (France).

The objective of the OZONE project is to specify and implement a generic architecture/framework that will support the effective acceptance and use of ambient intelligence in the consumer domain. The OZONE project aims at the development of novel concepts, techniques and tools to provide invisible computing for the domestic and nomadic personal use of information technology. The developed concepts aim at improving the acceptability and usability for the average customer. One of the important concepts is the application of advanced technologies to support the user-centric retrieval and consumption of information compared to the current-practice, computer-centric approach. This requires special emphasis on natural interfaces that put the user in the foreground and the system in the background. Security and privacy are prerequisites for consumer acceptance of these systems and are covered by the project's software environment. A final objective deals with the provision of a strong technology base enabling powerful, but energy-efficient, computing.

8.2. International Research Networks and Work Groups

8.2.1. IST NoE CaberNet

- **Name:** IST NoE CaberNet – Network of excellence in distributed and dependable systems
- **URL** : http://www.newcastle.research.ec.org/cabernet/
- **Period:** [January 2001 - March 2004]
- **Coordinating partners:** University of Newcastle Newcastle (UK) - Lead contractor, CNRS-LAAS (France), INRIA (UR Rocquencourt), Lancaster University (UK), Technical University of Vienna (Austria), University of Bologna (Italy), University of Cambridge (UK), University of Kaiserslautern (Germany), University of Lisbon (Portugal), University of Pisa (Italy), University of Twente (The Netherlands).

CaberNet co-ordinates top-ranking European research in distributed and dependable computing systems architectures. Distributed and dependable computing systems are fundamental to the successful development of the Information Society. Large distributed network infrastructures such as the Internet are vital for citizens to benefit from services such as the global market place, education and information, while preserving their rights to freedom of expression, privacy, intellectual property, etc. But this development depends mostly on how much users will trust the services offered to them. It is therefore essential to make these systems dependable.

8.2.2. IST WG iTrust

- **Name:** IST WG iTrust – Working group on trust management in dynamic open systems
- **URL:** http://www.itrust.uoc.gr/
- **Period:** [September 2002 - August 2005]
- **Partners:** University of Crete (Greece) – Project coordinator, CCLRC (UK), CNR-ISTC (Italy), HP (UK), Imperial College (UK), INRIA (UR Rocquencourt), Intracom SA (Greece), King's College (UK), Nine by Nive Co (UK), Plefsis Information Systems SA (Greece), Queen Mary University College (UK), Sintef telecom and Informatics (Norway), Trinity College Dublin (Ireland), Autonomous University of Barcelona (Spain), University of Dortmund (Germany), University of Oslo (Norway), University of Strathclyde (UK), Virtual Trip Ltd (Greece),

The aim of iTrust is to provide a forum for cross-disciplinary investigation of the application of trust as a means of establishing security and confidence in the global computing infrastructure, recognizing trust as a crucial enabler for meaningful and mutually beneficial interactions. The proposed forum brings together researchers with a keen interest in complementary aspects of trust, from both technology-oriented disciplines and the field of law, social sciences and philosophy. Hence, it aims at providing the consortium participants (and the research communities associated with them) with the common background necessary for advancing toward an in-depth understanding of the fundamental issues and challenges in the area of trust management in open systems.

8.3. Ministry Grant

8.3.1. ACI CorSS

- **Name:** ACI CorSS – A formal approach to the composition and refinement of system services
- **URL:** http://www.irit.fr/CORSS
- **Related activity:** §6.2.1
- **Period:** [September 2003 - August 2006]

- **Partners:** SVF FERIA (Toulouse) – Project coordinator, ARLES at INRIA-Rocquencourt, OBASCO/LOAC at Ecole des Mines de Nantes (Nantes), COMPOSE at INRIA/LABRI (Bordeaux), MOSEL at LORIA (Nancy).

The CorSS project is a joint work between teams from the system community and teams from the formal methods community. Its aim is to study development mechanisms for ensuring the safety of the system services that are to be certified. The underlying development concepts are refinement and composition. The project in particular investigates specific formalisms, well suited for the development of systems, as well as their needs in terms of refinement and composition. More specifically, the project considers features interaction for telecommunication software, the derivation of robust Web services, and the composition of basic OS kernel services for which it examines relevant composition techniques and proof methods.

9. Dissemination

9.1. Involvement within Scientific Community

9.1.1. Programme Committees

- V. Issarny is PC member of MDM'03: 4th International Conference on Mobile Data Management. January 2003, Melbourne, Australia.
- V. Issarny and N. Levy are PC members of WADS'03: ICSE'2003 Workshop on Architecting Dependable Systems. May 2003, Portland, USA.
- V. Issarny is PC member of MCM'03: 1st International ICDCS Workshop on Mobile Computing Middleware. May 2003, Providence, USA.
- V. Issarny is PC member of WWW'03: 12th International World Wide Web Conference. May 2003, Budapest, Hungary.
- V. Issarny is PC member of iTrust'03: 1st International Conference on Trust Management. May 2003, Heraklion, Greece.
- V. Issarny is PC member of MPAC'03: 1st International Workshop on Middleware for Pervasive and Ad Hoc Computing. June 2003, Rio de Janeiro, Brazil.
- V. Issarny is PC member of ESEC/FSE'03: 4th joint meeting of the European Software Engineering Conference and ACM SIGSOFT Symposium on the Foundations of Software Engineering. September 2003, Helsinki, Finland.
- V. Issarny is PC member of MCTA'03: International Workshop on Mobile Commerce Technologies and Applications. September 2003, Prague, Czech Republic.
- N. Georgantas is PC member of CFSE'03: *3ème Conférence française sur les systèmes d'exploitation*. October 2003, La Colle sur Loup, France.
- V. Issarny is PC member of LADC'03: 1st Latin-American Symposium on Dependable Computing. October 2003, Sao Paulo, Brazil.
- N. Levy is PC member of SoMeT'03: 2nd International Workshop on Lyee Methodology. October 2003, Stockholm, Sweden.
- V. Issarny is PC member of MDM'04: 5th IEEE International Conference on Mobile Data Management. January 2004, Berkeley, California, USA.
- N. Levy is PC member of SE'04: IASTED International Conference on Software Engineering. February 2004, Innsbruck, Austria.

- V. Issarny is PC member of MP2P'04: First International Workshop on Mobile Peer-to-Peer Computing. March 2004, Orlando, Florida, USA.
- V. Issarny is PC member of iTrust'04: 2nd International Conference on Trust Management. Oxford, UK, March 2004.
- V. Issarny is PC member of MP'04: International Workshop on Middleware Performance. April 2004, Phoenix, Arizona, USA.
- V. Issarny is PC member of ICSE'04: 26th International Conference on Software Engineering. May 2004, Edinburgh, UK.
- V. Issarny and N. Levy are PC members of EWSA'04: First European Workshop on Software Architecture (co-located with ICSE'04). May 2004, Edinburgh, UK.
- V. Issarny and N. Levy are PC members of WADS'04: ICSE & DSN'04 Workshop on Architecting Dependable Systems. May 2004, Edinburgh, UK and July 2004, Florence, Italy.
- V. Issarny is PC member of WICSA'04: 4th IEEE/IFIP Working Conference on Software Architecture (co-located with ECOOP 2004). June 2004, Oslo, Norway.
- N. Levy is PC member of AFADL'04: *Approches formelles dans l'assistance au développement de logiciels*. June 2004, Besançon, France.
- V. Issarny is PC member of ICDE'05: 21st International Conference on Data Engineering. April 2005, Tokyo, Japan.

9.1.2. Other activities

- V. Issarny has been vice-chair of the ACM SIGOPS from June 1999 until June 2003.
- V. Issarny is chair of the executive committee of the AIR&D consortium on Ambient Intelligence Research and Development (http://www.air-d.org).
- V. Issarny is chair of the CaberNet Research Coordination and Training Committee.
- V. Issarny is member of the 2003 *prix ASF de la recherche en systèmes* committee.

9.2. Teaching

- N. Levy is responsible of the AOD (*Architecture à Objets Distribués*) option of the DEA MISI of the University of Versailles Saint-Quentin en Yvelines.
- V. Issarny and N. Levy give the main lecture of the AOD option of the DEA MISI of the University of Versailles saint-Quentin en Yvelines.
- V. Issarny gave a lecture on "Software Architecture and Dependability" at SFM-03:SA: 3rd International School on Formal Methods for the Design of Computer, Communication and Software Systems: Software Architecture. September 2003, Bertinoro, Italy.
- N. Georgantas gives a half-semester course on "Middleware" for the final year of a five-year computer engineering degree at the *Ecole Supérieure d'Ingénierie Léonard de Vinci* of the *Pôle Universitaire Léonard de Vinci*.
- F. Tartanoglu gives a course on "Web services" at the *Licence professionnelle ISDRN (Intégrateur de Systèmes Distribués et Réseaux Numériques)*, IUT Velizy, University of Versailles Saint-Quentin-en-Yvelines.

9.3. Internships

During year 2003, members of the ARLES project-team supervised the work of the following student interns:

- R. Chibout, "Web services for mobile, wireless devices", Graduate Sudent intern, *DEA Informatique Distribué*, University of Paris-11.
- S. Ziane, "Dynamic composition of Web services and fault tolerance", Graduate Student intern, *DEA MISI*, University of Versailles Saint-Quentin-en-Yvelines.

9.4. Invited Conferences

Members of the ARLES project-team gave presentations at conferences and workshops, as listed in the publication section. They also gave the following talks:

- V. Issarny. “Dependability in the Web Services Architecture”. DSoS Open Workshop. March 2003, Vienna, Austria.
- M. Boulkenafed. "AdHocFS: Ad Hoc Distributed File System for Mobile Users". Seminar. April 2003, University College London, London, UK.
- V. Issarny. "Towards the Development of Ambient Intelligence Systems: Supporting Data Sharing in Mobile Ad hoc Systems". Seminar. May 2003, University of Ioanina, Ionina, Greece.
- F. Sailhan. "*Vers le développement de système pour l'intelligence ambiante : une solution au partage de données dans les systèmes mobiles ad hoc*". *Journées thèmes émergeants, Infrastructure pour Petits Objets Portables et sécurisés*. September 2003, Lille, France.
- M. Boulkenafed. "Distributed System for Ubiquitous Collaboration". 5th Plenary CaberNet Workshop. November 2003, Porto Santo, Portugal.
- M. Boulkenafed. "Data Access Management over Ad Hoc Networks". Seminar. December 2003, Trinity College Dublin, Dublin, Ireland.

10. Bibliography

Major publications by the team in recent years

[1] R. ALLEN, D. GARLAN. *A Formal Basis for Architectural Connection.* in « ACM Transactions on Software Engineering and Methodology », number 3, volume 6, 1997, pages 213-249.

[2] W. EMMERICH. *Engineering Distributed Objects.* J. Wiley & Sons, 2000.

[3] D. GARLAN, R. ALLEN, J. OCKLERBLOOM. *Architectural Mismatch: Why Reuse Is So Hard.* in « IEEE Software », 1994, pages 17-26.

[4] V. ISSARNY, C. KLOUKINAS, A. ZARRAS. *Systematic Aid for Developing Middleware Architectures.* in « Communications of the ACM », volume 45(6), 2002.

[5] J. C. LAPRIE, editor, *Dependability: Basic Concepts and Terminology.* series Dependable Computing and Fault-Tolerant Systems, volume 5, Springer-Verlag, 1992.

[6] N. MEDVIDOVIC, R. N. TAYLOR. *A Classification and Comparison Framework for Software Architecture Description Languages.* in « IEEE Transactions on Software Engineering », number 1, volume 26, 2000.

[7] M. SHAW, D. GARLAN. *Software Architecture: Perspectives on an Emerging Disciplines.* Prentice Hall, 1996.

[8] D. B. TERRY, M. M. THEIMER, K. P. A. J. DEMERS, M. J. SPREITZER, C. H. HAUSER. *Managing Update Conflicts in a Weakly Connected Replicated Storage System.* in « Proceedings of the Fifteenth ACM Symposium on Operating Systems Principles », 1995.

Doctoral dissertations and "Habilitation" theses

[9] M. BOULKENAFED. *Gestion de l'accès aux données dans les réseaux mobiles sans fil.* Thèse de doctorat, University of Paris 6, November, 2003.

Articles in referred journals and book chapters

[10] T. HIGUERA, V. ISSARNY, M. BANÂTRE, G. CABILLIC, F. PARAIN, J. LESOT. *Memory Management for Real-time Java: an Efficient Solution using Hardware Support.* in « Real-Time Systems journal (Kluwer Pub.) », 2004, to appear.

[11] 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 « Journal of Automated Software Engineering », 2004, to appear.

[12] V. ISSARNY, A. ZARRAS. *Formal Methods for Software Architectures - SFM-03:SA Lectures.* LNCS 2804, Springer, September, 2003, chapter Software Architecture and Dependability.

[13] F. SAILHAN, V. ISSARNY. *Handbook of Mobile Computing.* CRC Press, 2004, chapter Energy-aware Web Caching over Hybrid Networks, to appear.

[14] F. TARTANOGLU, V. ISSARNY, A. ROMANOVSKY, N. LEVY. *Architecting Dependable Systems.* LNCS 2677, Springer, June, 2003, chapter Dependability in the Web Services Architecture.

[15] A. ZARRAS, C. KLOUKINAS, V. ISSARNY. *Architecting Dependable Systems.* LNCS 2677, Springer, June, 2003, chapter Quality Analysis of Dependable Systems: A Developer Oriented Approach.

Publications in Conferences and Workshops

[16] M. BOULKENAFED, V. ISSARNY. *A Middleware Service for Mobile Ad Hoc Data Sharing, Enhancing Data Availability.* in « Proc. of the 4th ACM/IFIP/USENIX International Middleware Conference », Rio de Janeiro, Brazil, June, 2003.

[17] M. BOULKENAFED, V. ISSARNY. *AdHocFS: Sharing Files in WLANs.* in « Proc. of the 2nd International Symposium on Network Computing and Applications », Cambridge, MA, USA, April, 2003.

[18] J. LIU, V. ISSARNY. *QoS-aware Service Location in Mobile Ad Hoc Networks.* in « Proc. of the 5th IEEE International Conference on Mobile Data Management », Berkeley, CA, USA, January, 2004, to appear.

[19] F. SAILHAN, V. ISSARNY. *Cooperative Caching in Ad Hoc Networks.* in « Proc. of the 4th International Conference on Mobile Data Management », Melbourne, Australie, January, 2003.

[20] F. TARTANOGLU, V. ISSARNY, N. LEVY, A. ROMANOVSKY. *Formalizing Dependability Mechanisms in B.* in « Proc. of the ICSE'2003 Workshop on Architecting Dependable Systems », Portland, OR, USA, May, 2003.

[21] F. TARTANOGLU, V. ISSARNY, A. ROMANOVSKY, N. LEVY. *Coordinated Forward Error Recovery for Composite Web Services.* in « Proc. of the 22nd Symposium on Reliable Distributed Systems (SRDS'2003) », Pisa, Italy, October, 2003.

Internal Reports

[22] R. BHASKAR. *Group Key Agreement in Ad Hoc Networks.* Technical report, number 4832, INRIA, May, 2003, http://www.inria.fr/rrrt/rr-4832.html.

[23] M. BOULKENAFED, J. LIU, D. SACCHETTI, V. ISSARNY. *Group Management in Mobile Ad Hoc Networks: Design, Implementation and Experiments.* Technical report, number 5060, INRIA, December, 2003, http://www.inria.fr/rrrt/rr-5060.html.

[24] N. GEORGANTAS, D. SACCHETTI, V. ISSARNY. *Making Middleware Communication Architecture Reconfigurable.* Technical report, number 0279, INRIA, April, 2003, http://www.inria.fr/rrrt/rt-0279.html.

Miscellaneous

[25] *D10: Interim Report on the Integration Process and First Demonstrable Integrated OZONE Prototype and Demonstrators.* OZONE Project Deliverable, October, 2003.

[26] *D8b: Intermediate Status Report on the Implementation of the Software Environment and First Demonstrable Prototype.* OZONE Project Deliverable, July, 2003.

[27] *D8: Intermediate Status Report on the Implementation.* OZONE Project Deliverable, July, 2003.

[28] *D9a: Application Scenarios.* OZONE Project Deliverable, October, 2003.

[29] F. TARTANOGLU, M.-C. GAUDEL, N. GEORGANTAS, V. ISSARNY, N. LEVY, A. ROMANOVSKY. *Architecting DSoSs using the Web Services Architecture.* Chapter of DSoS Project Deliverable, March, 2003.