INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team aces

Ambient Computing and Embedded Systems

Rennes

THEME COM

Table of contents

1. Team

Head of project-team
Michel Banâtre [Research Director, INRIA, HdR]

Administrative assistant
Evelyne Livache

INRIA project staff
Ciaran Bryce [Research Director, INRIA]
Paul Couderc [Research Scientist, INRIA]

CNRS project staff
Jean-Paul Routeau [Senior technical staff, CNRS]

University project staff
Frédéric Weis [Assistant Professor, IUT de Saint-Malo]

Project technical staff
Fabien Allard
Mathieu Becus
Ronan Menard [since 1/2/2006]
Antoine Luu [up to 31/8/2006]

Ph. D. student
Damien Martin-Gutteriez [ENS grant]
Mickaël LeBaillif [Inria grant]
Xavier Le Bourdon [MESR grant]
Pierre Duquesne [INRIA grant, since 1/10/2006]
Azza Jediddi [INRIA grant, since 1/10/2006]
Claude Vittoria [Inria grant]
Cedric Mosch [Inria grant]
Arnaud Guiton [Inria grant]
Mazen Tlais [Inria grant]

2. Overall Objectives

2.1. Overall Objectives

Three key phenomena have been changing the nature of computing over the last few years. The first is the popularity of portable devices such as mobile telephones and Personal Digital Assistants (PDAs). Today, around 80% of the French adult population possess their own mobile phone and there is a large variety of smartphones on the market that integrate PDA functionality. The second phenomenon is the large number of embedded systems; these are everyday devices that have their own processor and memory. Estimates suggest that more than 98% of the world's processor's are in embedded systems [19], thus facilitating the deployment of a variety of information systems that control physical objects. The third phenomena is the increasing variety of wireless networks available for personal and embedded devices, e.g., Bluetooth, Wifi, GPRS, etc.

The combination of these three phenomena has permitted the emergence of context-aware person-centric applications and services. These services complement a person's physical ability to interact with his environment. They are tailored to a the needs, preferences and location of each person carrying a device, and are continually available. Services range from critical, e.g., remote health monitoring [22], to utility, e.g., navigational help, etc. to value-added, e.g., virtual museum guides, smart home, etc.

The domain of person-centric computing is known in research circles as *ambient computing* [25], and several significant research challenges remain. First, to facilitate mobility, ambient computing services should require minimal device manipulation by the device owner. It is crucial that the computing device operate as an extension of the person rather than as a tool. Second, there must be a way of modeling the physical environment so that applications can seamlessly import data from the environment and modify the environment when possible. Third, applications must be able to adapt to the rather limited storage and processing capabilities of mobile devices, as well as to variable and intermittent wireless network coverage.

The ACES (Ambient Computing and Embedded Systems) group is addressing research from three angles[1]:

- *Programming Models for Ambient Computing*. We have looked at ways of modeling the physical environment in the virtual environment of programs in order to facilitate ambient application development. The goal is to be able to write programs that address and navigate through objects in the physical world as elegantly as a program traditionally manipulates a computer's main memory.
- *Quality of Ambient Service*. A user needs to be able to exploit ambient services as seamlessly as possible. In particular, he should be shielded from the effects of network breaks – something that can be quite common for wireless environments. We have been developing for ensuring continuous video streaming for mobile devices.
- *Operating System Support*. We are looking at portable operating system designs for personal devices that manage limited physical resources and irregular network connectivity.

Throughout 2006, in accordance with the goals of the institute, the ACES group spent a great deal of time and effort on seeking to transfer its research results on ambient computing to potential users. We sought out and negotiated with representatives of end-users in advertising and the urban construction industries. Our motivation stems from our observation that producing innovative research results, even those protected by patents, is no longer sufficient for a modern research team. It is essential to convince industry that solutions are robust, scalable and most importantly, address a problem that real users are faced with. This research approach necessitates the development of several prototypes that are tested in real environments. It also necessitates a continuous technology watch to ensure the validity of submitted patents, as well as verification of existing patents and research reports. Although this activity is, traditionally, unusual for a research team, it becomes inevitable if results are to have a real impact in the ambient computing applications currently being deployed.

Our technology transfer efforts have been very successful. A start-up company called UbiSphere went into INRIA incubator status in October 2006. A major patent was transferred to JCDecaux – the worldwide billboard operator – in February 2006. From November 2006 to February 2007, the group has been commissioned by ADP to design and implement a pilot of an ambient system, based on the group's technology, for use in airport environments.

Despite our work with industrial partners, we have not neglected academic partners. For instance, we are participating in Mosaic – a research project financed by the French government. At a European level, we are participating in the Resist network of Excellence, the Wisents coordinated action and, since November 2006, the Roboswarm EU project.

This document overviews our activities in more detail. The section *Scientific Foundations* gives some background to our work in person-centric computing. The section *Application Domains* describes the importance of our research agenda through the presentation of several applications, some of which are being developed in our group. The group's recent results are presented in the section *New Results*.

3. Scientific Foundations

3.1. Introduction

Keywords: *Embedded systems*, *ambient computing*, *design tools*, *energy consumption*, *hard/soft real-time*, *spatial navigation*, *ubiquitous computing*.

[1] At the outset of the four-year period, we also had real-time system support as an objective. However, this research theme ended when Isabelle Puaut left the group in 2004.

The following paragraphs give a quick overview of the scientific background of the ACES research activities. Ambient computing and embedded systems are the foundations of person-centric computing. Our group is concentrating on *programming models* and *operating system support*; these are two essential and complementary aspects of ambient computing.

The purpose of a programming model is to represent information as data, and to provide a computational framework for data processing. The challenge for ambient and embedded computing is to seamlessly merge information from the physical and virtual worlds, so that programs can act upon and influence the physical world around them.

The goal of our research on operating system support is to define platforms that enable programs to run on resource-limited devices, where wireless network connection fluctuates. In particular, we are looking at environments built around Java^{TM} technology, in order to maximise application safety and portability.

3.2. Programming Models

The goal of ambient computing is to seamlessly merge virtual and real environments. A real environment is composed of objects from the physical world, e.g., people, places, machines. A virtual environment is any information system, e.g., the Web. The integration of these environments must permit people and their information systems to implicitly interact with their surrounding environment.

Ambient computing applications are able to evaluate the state of the real world through sensing technologies. This information can include the position of a person (caught with a localisation system like GPS), the weather (captured using specialised sensors), etc. Sensing technologies enable applications to automatically update digital information about events or entities in the physical world. Further, interfaces can be used to act on the physical world based on information processed in the digital environment. For example, the windows of a car can be automatically closed when it is raining.

This real-world and virtual-world integration must permit people to implicitly interact with their surrounding environment. This means that manual device manipulation must be minimal since this constrains person mobility. In any case, the relative small size of personal devices can make them awkward to manipulate. In the near future, interaction must be possible without people being aware of the presence of neighbouring processors.

3.2.1. Programming Context

Information systems require tools to *capture* data in its physical environment, and then to *interpret*, or process, this data. A context denotes all information that is pertinent to a person-centric application. There are three classes of context information:

- The *digital context* defines all parameters related to the hardware and software configuration of the device. Examples include the presence (or absence) of a network, the available bandwidth, the connected peripherals (printer, screen), storage capacity, CPU power, available executables, etc.
- The *personal context* defines all parameters related to the identity, preferences and location of the person who owns the device. This context is important for deciding the type of information that a personal device needs to acquire at any given moment.
- The *physical context* relates to the person's environment; this includes climatic condition, noise level, luminosity, as well as date and time.

All three forms of context are fundamental to person-centric computing. Consider for instance a virtual museum guide service that is offered via a PDA. Each visitor has his own PDA that permits him to receive and visualise information about surrounding artworks. In this application, the *pertinent* context of the person is made up of the artworks situated near the person, the artworks that interest him as well as the degree of specialisation of the information, i.e., if the person is an art expert, he will desire more detail than the occasional museum visitor.

There are two approaches to organising data in a real to virtual world mapping: a so-called *logical* approach and a *physical* approach. The logical approach is the traditional way, and involves storing all data relevant to the physical world on a service platform such as a centralised database. Context information is sent to a person in response to a request containing the person's location co-ordinates and preferences. In the example of the virtual museum guide, a person's device transmits its location to the server, which replies with descriptions of neighbouring artworks.

The main drawbacks of this approach are scalability and complexity. Scalability is a problem since we are evolving towards a world with billions of embedded devices; complexity is a problem since the majority of physical objects are unrelated, and no management body can cater for the integration of their data into a service platform. Further, the model of the physical world must be up to date, so the more dynamic a system is, the more updates are needed. The services platform quickly becomes a potential bottleneck if it must deliver services to all people.

The physical approach does not rely on a digital model of the physical world. The service is computed wherever the person is located. This is done by spreading data onto the devices in the physical environment; there are a sufficient number of embedded systems with wireless transceivers around to support this approach. Each device manages and stores the data of its associated object. In this way, data are physically linked to objects, and there is no need to update a positional database when physical objects move since the data *physically* moves with them.

With the physical approach, computations are done on the personal and available embedded devices. Devices interact when they are within communication range. The interactions constitute delivery of service to the person. Returning to the museum example, data is directly embedded in a painting's frame. When the visitor's guide meets (connects) to a painting's devices, it receives the information about the painting and displays it.

3.2.2. Spatial Information Systems

One of the major research efforts in ACES over the last few years has been the definition of the Spread programming model to cater for spacial context. The model is derived from the Linda [21] tuple-space model. Each information item is a *tuple*, which is a sequence of typed data items. For example, `<10, 'Peter', -3.14>` is a tuple where the first element is the integer 10, the second is the string ‘"Peter" and the third is the real value -3.14. Information is addressed using patterns that match one or a set of tuples present in the tuple-space. An example pattern that matches the previous tuple is `<int, 'Peter', float>`. The tuple-space model has the advantage of allowing devices that meet for the first time to exchange data since there is no notion of names or addresses.

Data items are not only addressed by their type, but also by the physical space in which they reside. The size of the space is determined by the strength of the radio signal of the device. The important difference between Spread and other tuple-space systems (e.g., Sun's JavaSpaces [20], IBM's T-Space [28]) is that when a program issues a matching request, only the tuples filling the *physical space* of the requesting program are tested for matching. Thus, though applications are highly distributed by nature, they only rely on localised communications; they do not require access to a global communication infrastructure. Figure 1 shows an example of a physical tuple space, made of tuples arranged in the space and occupying different spaces.

As an example of the power of this model, consider two of the applications that we have developed using it.

- *Ubi-bus* is a spatial information application whose role is to help blind and partially blind people use public transport. When taking a bus, a blind person uses his PDA to signal his intention to a device embedded in the bus stop; this device then contacts the bus on the person's behalf. This application illustrates how data is distributed over the objects of the physical world, and generally, how devices complement human means of communication.
- *Ubi-board* is a spatial information application designed for public electronic billboards. Travel hotspots like airports and major train stations have an international customer base, so bill-board announcements need to be made in several languages. In Ubi-bus, a billboard has an embedded device. When a person comes within communication range of the billboard, his device sends a

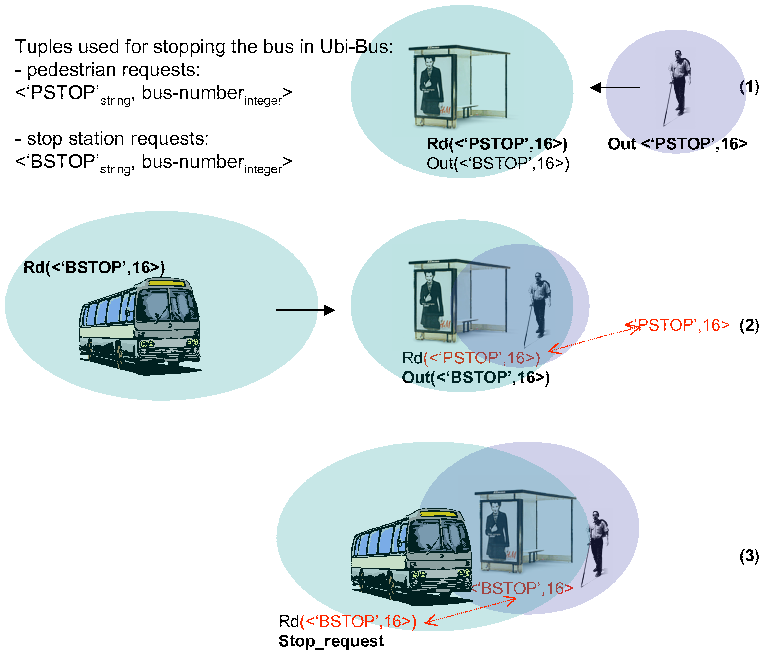

Figure 1. Physical Tuple Space

request to the billboard asking it to print the message in the language of the person. In the case where several travellers are in proximity of the billboard, the board sends a translation of its information message to each person. The Ubi-board application illustrates personal context in use, i.e., the choice of natural language, and also how actions can be provoked in the physical world without explicit intervention by the person.

3.3. Operating System Support

The role of an operating system is to offer an environment for program execution. It offers programs access to essential services, such as communication and storage, and thus indirectly to network and disk. Person-centric computing is characterised by small portable devices that can be awkward for a person to manipulate. The operating system needs to manage generally limited resource, in particularly in relation to the network. We first give some background to the problem of resource management – given its importance in person-centric computing – before coming back to other operating system aspects.

3.3.1. Service Adaptation

Mobile networks are becoming increasingly heterogeneous. Global coverage is now well provided by 2G and 2.5G cellular systems, and 3G networks (UTMS) are being deployed in some densely populated areas. Nonetheless, high data rates (several Mb/s) will not be available everywhere in the near future, so the delivery of large amounts of information to people on the move will remain limited and expensive.

3.3.1.1. Data Adaptation

Wireless network coverage in mobile systems can be highly variable for reasons of coverage as well as for reasons related to subscription business models. This variability suggests that an application must be able to adapt to network related changes. An application must also be able to adapt to particular resource limitations of devices it communicates with at any time.

Ultimately, adaptation manifests itself in the quality and nature of data exchanged between devices. One solution to adapting data is to transform its representation in a way that best suits the digital context of host devices.

For example, depending on the actual behaviour of the system (network bandwidth, load, etc.), it implies the ability to manipulate different representations of the same data that are physically different but semantically equivalent (e.g., the color and black-and-white versions of the same picture). These representations differ only in quality. For instance, when transmitting video to a mobile device, the number of frames per second can be reduced to cater for variable bandwidth, and the size of each frame can be significantly reduced to cater for the fact that the screen is quite small. Another example is Web browsing; a mobile device might decide to filter applets and images from downloaded pages in order to reduce the device CPU time needed to process the page.

Data adaptability requires collaboration between the operating system and the applications [23]. As the previous examples illustrate, the type of data adaptation relies on the semantics of the application, so only it can decide how to adapt to resource changes. It is the role of the operating system to survey changes in resources, and to inform the application of important changes so that adaptation can take place in time.

An unstable connection can temporarily isolate a mobile entity from the rest of the system. When this happens, the proposed service cannot be properly delivered. In order to hide disconnections from a person, pre-fetching techniques can be used: documents are pre-loaded in a local cache of the mobile device during high connectivity periods. The application therefore remains operational when a temporal disconnection occurs as all needed documents have been stored locally in the cache of the device. The data loaded into the caches are selected based on the personal context, incorporating a person's preferences, as well as on hints provided by the user and on information that is automatically tracked (for example a log of all previously accessed documents).

3.3.1.2. The Pico-cell Architecture

The past few years have witnessed the rise of the cellular networks. These communication systems were designed with a philosophy of *any-time any-where* service. Users wish to receive and place calls at any location and without delay, to move while talking without interrupting their conversations. This requires ubiquitous coverage, which in turn requires significant infrastructure. A modern cellular system is installed with hundreds of base stations, at a cost of hundreds of millions of euros, in order that a communication link is always available. Such any-time any-where service provision becomes increasingly expensive and suffers from low bandwidth. Covering wide areas with high radio bandwidth requires complex equalization, due to signal attenuation, multi-path fade, and shadowing effects. Sophisticated radio engineering will lead to improved bandwidth, coverage, and mobile access, but this will be expensive, in terms of both capabilities and cost.

In this context, the ACES project has studied an alternate design for wireless networks where intermittent but very high speed is provided to the network through *Pico-cells*. The latter consists of a set of access points (APs)-based for example on WLANs cells-, *i.e.* antennas around of which are defined radio cells with limited range (about 100 meters). Those antennas are discontinuously spread on the network area, thus providing a *many-time many-where* service. Actually, the idea of coverage discontinuity brings two major advantages. First, as it implies the use of a fewer number of access points, the architecture deployment will be cheaper. Second, radio cells disjunction hypothesis simplifies the radio frequency band management and avoids interference problems.

Even if this model simplifies network deployment, the connectivity intermittence induces important challenges in order to avoid service disruption. Thus, terminals have to take advantage of the high bandwidth when it is available. For delay tolerant data, a terminal stores data as it passes under a cell. Hence, it may consume the buffered data even when it passes through regions of poor network coverage. Many projects have studied very specific cases where the cells deployment is uniform and data is sent from servers to terminals (down-link). One example of this type of system is studied by WINLAB (Wireless Information Network Laboratory). In this project, cells are equally spaced and the data delivery algorithm is tested in a network with one dimensional system *i.e.* high ways.

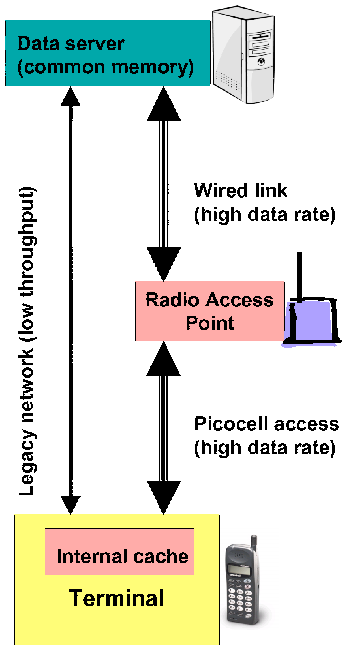

Figure 2. Pico-cell architecture model

Our approach is based on a more general case in which cells are distributed according to the envisaged traffic and data can be exchanged in both directions down-link (from servers to terminals) and up-link (from terminals to servers). The main challenge is to provide system mechanisms and efficient services addressing the specific constraints of this architecture: discontinuous coverage, user unconstrained mobility, high user density. To cope with the discontinuous coverage of the network, we store data (with caching mechanisms) close to mobile people, just before data delivery. Thus, the placement policy of data within the architecture is conditioned by knowledge of people on the move. The goal here is to define a representation of person mobility in the network architecture, and to use this model for placing data using limited and customised flooding mechanisms.

Through this architecture model, we underline the analogy between heterogeneous mobile networks and multiprocessor architectures (for example the mobile device can be considered as a processor). This approach allows us to map and extend existing caching mechanisms, taking into account the specific constraints of a discontinuous mobile network. This architecture and the attached mechanisms have been evaluated with a simulation platform (See section on *Software*).

3.3.2. Operating Systems for Small Devices

Recent years have seen a huge proliferation of applications developed in the Java^{TM} programming language [17]. The advantage of Java is that the language is strongly typed, so programs are protected from the memory manipulation errors that are frequent with C/C++ applications. Another reason for its success is that Java environments now possess a rich set of libraries that are convenient for a wide range of systems programming tasks. A final, and important, reason for the prevalence of Java is that the Java compiler transforms programs to a standardised set of high-level assembly-like instructions called *bytecode*. Java environments incorporate a virtual machine that interprets bytecode. The advantage of this is portability: any platform that runs a Java environment may run a compiled Java program, without modification. Java

environments now exists for mobile phones, pagers, PDAs and a wide range of embedded and wireless devices[2].

3.3.2.1. Java Operating Systems

The ACES research group has been collaborating with *Texas Instruments* on the design of a heterogeneous multiprocessor architecture composed of a dedicated Java bytecode processor (JSM) and ARM processor. This work led us to design an operating system that is completely written in the Java programming language, and that can run on any abstract Java bytecode processor platform.

The advantage of a Java operating system running on an abstract bytecode processor is portability and safety. Currently, Java environments permit applications to interact with non-Java code, which is generally referred to as *native* code. Native code is used to implement operating system services, drivers as well as legacy applications. The problem with native code is that it creates an application dependency on third-party non-Java code. This challenges application portability since an application on one platform is no longer guaranteed to run on another platform. Further, native code is not subject to the type-safety checks of the Java language; this code can therefore compromise the security of the whole application.

There have been several R & D efforts to develop Java operating systems and Java environments written in the Java language, e.g., JX [24], JNode[3], OVM[4], Jalapeño [16], etc. These systems still contain native code units. JNode and Jalapeño compile the system's code base to native code for a host platform processor and OS pair in an effort to maximise run-time performance. However, this approach means that no use can be made of any bytecode processor, and the system requires extra means to protect the Java environment from all imported native code.

The idea of a machine that directly implements a language is not new, with many attempts to build such architectures in the 1970s, e.g., the Burroughs machine [18] designed to run Algol-60 [26] and the SAR designed to support Algol-68. Other notable efforts include Hydra [27] from CMU and the iAPX432 from Intel [15]. The goal of these projects was to run a programming language directly on a hardware architecture. This approach provides the obvious benefits of having the programming language semantics directly implemented by the hardware. However, these systems had three important drawbacks. First, the implementation of communication channels to interface between the programming language and the underlying hardware was complicated. Second, performance was generally poor. Third, the programming languages chosen lacked the popularity needed for the machines to reach a critical mass. The Java language does not suffer from the popularity drawback, and the availability of JSM processors that execute bytecode is helping to address the performance issue. The remaining challenge is program-to-hardware communication, and is an issue being addressed in the ACES research group.

3.3.2.2. Native Objects and the Java Abstract Machine

A portable Java operating system environment is one where all programs, services and operating system functions are compiled to bytecode. The only non-Java, or complementary instruction set architecture (C-ISA), instructions are those that manipulate attached hardware resources, e.g., keyboard, screen, communication ports, etc. The ACES group developed an operating system model in which devices are represented as Java objects whose methods can only be invoked by the operating system code (written in Java). These objects are known as *native objects*. Their methods are implemented independently of the Java layers: in microcode for hardware implementations of the abstract processor, in C or even Java for software implementations.

The structure of our Java operating system is illustrated in Figure 3. The native objects are placed at the same level as the abstract bytecode processor. To date, we have developed native objects for a keyboard, video card and UART serial bus. As with any processor, the bytecode processor supports the deployment of interrupt handlers so that the operating system services can manage devices. The next layer is composed of two parts. A Java services part implements all functionality needed to implement Java programs that is is not provided by the bytecode; this includes garbage collection of unused objects, thread scheduling and code

[2] http://java.sun.com/j2me/
[3] http://www.jnode.org
[4] http://www.ovmj.org

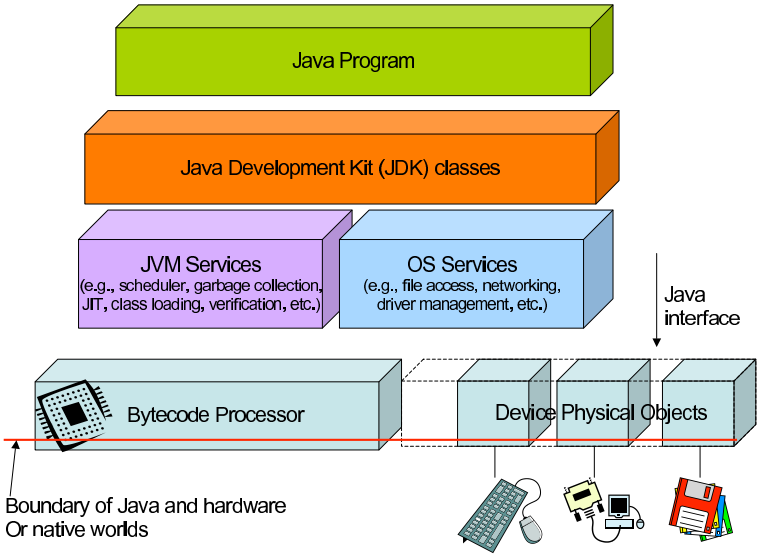

Figure 3. Structure of Java-based Operating System

verification of classes loaded into the system. The operating system part contains the device drivers and other OS functionality written in Java; for instance, we ported a TCP/IP stack and LCUDI graphics library to Java for integration in this part. The remaining two layers are common to all Java environments. The presence of the Java development kit (JDK) means that Java applications can run unaware of the fact that a bytecode processor is running below them. In particular, user code and libraries cannot invoke methods on native objects.

Though our Java operating system is designed to run over a physical bytecode processor, it can just as easily run over a software emulation of this processor in native code. In this respect the bytecode processor is an example of an *abstract machine*. Implementing this abstract machine is simpler than implementing a traditional Java virtual machine since only the bytecode interpretation component needs to be implemented instead of a whole series of thread and memory management services. Further, this implementation is hidden from all application and system service developers who can in no way interfere with the correct functioning of the abstract machine, since their code is written in Java. Thus, a native implementation of the abstract machine does not suffer from the portability and security shortfalls that we mentioned above for existing Java environments.

We return to our work on Java operating systems in Section *New Results* where we detail the recent innovative elements of our work .

4. Software

4.1. Introduction

The research tasks conducted in the ACES project lead to the development of many softwares. These developments are mainly realized, or at least initialized within the framework of industrial collaborations, and so they are attached to the application domains covered by the project.

4.2. Simulation Platform for Networks with Discontinuous Coverage

The ACES group developed a network simulator for pico-cell architectures that is used to analyse flow distribution in networks. This architecture is described in the *New Results* section of this report. The simulator is entirely coded in the Java programming language and uses the DESMO-J discrete event model. The simulator models all entities required to study discontinuous coverage network behaviour, including terminals, access points, admission control, content provision servers, MAC layers, wireless transmission, protocol cache management as well as different (person) device mobility models. The main emphasis of our development has been to tailor the simulator to measuring performance in large-scale networks – the size of towns where there are hundreds of devices per square kilometers – over periods of a few hours.

4.3. Ubi-Board

Ubi-Board is a splitted display system, capable of adapting contextually the content and the display surface accordingly to the population in the immediate vincinity of a display panel. The system uses spatial programming in order to determine dominant profiles, such as dominant language which allows the selection of the appropriate content to display on the main panel. As the main panel changes, the system synchronously pushes the appropriate content version to users not belonging to the dominant profile, to be displayed on their personal display (typically cellphones).

5. New Results

5.1. Introduction

The ACES project is currently very active in three main research activities

- Programming models
- Discontinuous mobile networks
- Java Operating Systems

In the following we give the major research results we got from these activities.

5.2. Programming models

Participants: Michel Banâtre, Mathieu Bécus, Paul Couderc [contact], Damien Martin-Gutteriez, Mickael LeBaillif, Xavier Le Bourdon

The ambient computing activity for this year include three main topics, that we describe in the next sections.

5.2.1. Geometry and pervasive computing

The role of geometry is important in pervasive computing as we can derive implicit data organization from spatial properties, and implicit data processing from mobility. Spatial programming is a computation model using this approach, allowing simple programming of pervasive computing application distributed over a set of physical objects. A Ph. D was completed on this topic, and has be defended by Julien Pauty in April 2006 [6].

A result of this work is a solution for the problem of atomic operation in the context of spontaneous communications, such as for example the atomic token passing between two mobile devices. This problem has no general solution, due to the inherently unreliable nature of wireless communication and the limited communication time implied by device mobility. However, partial solutions to the problem are proposed, involving a modified two phase commit protocol where engagement is restricted by a geometrical constraint. This constraint reduces the zone in which the protocol can start, in order to allow the completion before the devices be out of reach. We have shown that although it is not possible to guarantee failure free operation, acceptable reliability can be achieved for certain applications.

5.2.2. Context sensitive systems

The notion of context as it is used in most ubiquitous computing work is strongly different of the notion of context used in information system. Typically, context is understood by the ubiquitous computing community as a set of information charaterizing a physical situation. The notion of context in information systems corresponds to the relative location of a given information item, in respect to a more global set of information. We introduced in the past a unified definition of context, based on the definition used in information system, but also covering physical aspects. In this definition, we consider the physical space as an implicit structure for information associated to physical objects or region. This definition provides a support for organizing and navigating an information collectionimplicitely from the physical space.

Implementing this definition relies on sensing technologies to collect information from physical objects. We are now using RFID, in combination with other technologies such as Bluetooth and WLAN that we used in in the past. RFID provides the advantage to allow precise spatial sensing, hence allowing more accurate geometrical definition for context settings. These new architectures will be experimented in a collaboration with Aeroports de Paris : Ubi Board, a context sensitive display system that determines the "dominant" profile in its viciny, accordingly to the current majority. This is used to switch the content of the display in the appropriate language (of the majority), but at the same time sending localized version of the content to mobile of each user in a minority.

5.2.3. Cooperating objects

The third topic is the cooperation of autonomous objects. With today proliferation of mobile devices enabled with communication and multimedia data capture capabilities, large quantity of data production and processing directly takes place on the move, or in situation. Heterogenous communication interface and node architecture, interactions with unknown nodes (hence, under limited trust), data reliability, global processing over a set of nodes are some of the challenging issues raised by cooperating objects. Wireless sensor networks have similar issues, in addition to exacerbated energy and resources constraints.

We were involved in the Embedded WiSents European cooperation, as a study leader regarding programming abstractions and system architecture for cooperating objects.

Key to the successful and widespread deployment of cooperating objects and sensor network technologies is the provision of appropriate programming abstractions and the establishment of efficient system architectures able to deal with the complexity of such systems. Programming abstractions shield the programmer from the “system specific details” and allow the developer to think in terms of the concrete application problem rather then in terms of the system. This is also true for traditional distributed systems, where numerous software frameworks and middleware architectures are crucial to perform an integrated computing task. Such frameworks and middleware are based on programming models such as distributed objects or events. These conventional and successful distributed programming abstractions can, however, not be simply applied to cooperating objects or sensor networks, due to the differences existing among the latter and the former systems.

We are also involved in another cooperation, the french ACI Mosaic, which study the problem of cooperative backup of mobile devices. The first objective is to define an automatic data back-up and recovery service based on mutual cooperation between mobile devices with no prior trust relationships. Such a service aims to ensure continuous availability of critical data managed by mobile devices that are particularly prone to energy depletion, physical damage, loss or theft. The basic idea is to allow a mobile device to exploit accessible peer devices to manage backups of its critical data. The implementation of such a service by cooperation between devices with no prior trust relationship is far from trivial since new threats are introduced: (a) selfish devices may refuse to cooperate, (b) backup repository devices may themselves fail or attack the confidentiality or integrity of the backup data; (c) rogue devices may seek to deny service to peer devices by flooding them with fake backup requests; etc [9].

Dealing with these threats is the second objective of the project. We are studying mechanisms for managing trust in cooperative services between mutually suspicious devices. Of particular interest are mechanisms based on reputation (for a priori confidence-rating and a posteriori accountability) and rewards (for cooperation

incitation). In the sparse ephemeral networks of devices considered, these mechanisms can rely neither on accessibility to trusted third parties nor on connectivity of a majority of the considered population of devices. Self-carried reputation and rewards are therefore of prime interest.

In this study, we are focused on the energy issues of the backup protocol, and on analysing the data production and use pattern of typical mobile applications.

Two new perspectives around cooperating objects are being developed now - Cooperation of robots, where the physical space is tagged with instructions and/or data. We participate to a new European project, Roboswarm, on this research theme. - Spontaneous collaboration of capture devices (such as video/sound sensors), where mobile devices improve the reliability and quality of their local capture capabilities by collaborating with other nodes.

5.3. Discontinous Mobile Networks

Participants: Fabien Allard, Carole Bonan, Azza Jedidi, Antoine Luu, Ronan Ménard, Mazen Tlais, Frédéric Weis [contact]

During this past year, we designed an architecture capable of supporting video streaming over a pico-cell type infrastructure. The core idea is to insert a new component, called an *access controller*, that acts as a connector between the IP infrastructure and high-bandwidth cells. The access controller is able to efficiently schedule (distribute) streams to different mobile nodes. Recently, we have extended this mechanism so that video streams are distinguished from less time-critical applications, e.g., Web, file transfer, etc.

5.3.1. Continuous data delivery

Our first goal consists in designing a scalable unicast streaming service for discontinuous networks. On the one hand, streaming servers are usually located in classical IP networks. Such networks are submitted to many constraints, like bandwidth variations. On the other hand, mobile terminals evolve in wireless discontinuous coverage areas, constituted of several access points. The first idea we propose in the context of such networks, is to design and introduce an intermediate equipment that will catch the video flows sent by the content server, and then efficiently distribute (*i.e.*schedule) them to mobile terminals in the network according to their radio conditions. This equipment is called the **Access Controller** (AC).

The AC acts as a cache which temporarily stores the part of the flow to be delivered. More precisely the AC is provided with many caches. Each of them is related with a single streaming request and contains the part of the flow to be sent to the terminal. A terminals also contains a cache to store some parts of the flow, in order to consume it while crossing out of coverage areas, and to avoid service disruption. And in the AC, a scheduling policy is implemented for classifying flows in priority queues, depending (1) on terminal positions in the radio cells and on (2) terminal cache contents. Simulations have highlighted the efficiency of the scheduling policy implemented in the AC.

Based on this architecture, a thesis has been defended by Carole Bonan on December 2006 [5].

5.3.2. Improving wireless network capacity by introducing logical discontinuous coverage

We also showed that the mechanisms defined initially in the framework of discontinuous coverage networks can be easily extended to the context of classical continuous coverage networks (like for example 3rd generation UMTS networks). In such networks, available radio bandwidth is highly varying, depending on terminal positions. *High bandwidth areas* are similar to Pico-cells coverage areas, and *low bandwidth areas* can be assimilated to previous out-of-coverage areas.

Actually, our idea is to insert into a continuous network an equipment called **anchor point**. The latter acts as an AC: its goal is to favour data transfers when terminals are located in high bandwidth areas. We have demonstrated through simulations that avoiding data transfers in low bandwidth areas improves system scalability. For that we have evaluated the number of service disruptions according to users density. The result is that the network capacity (a fixed users density without service disruption) is improved of about 300 % (see figure 4 when an anchor point is present in the network [13].

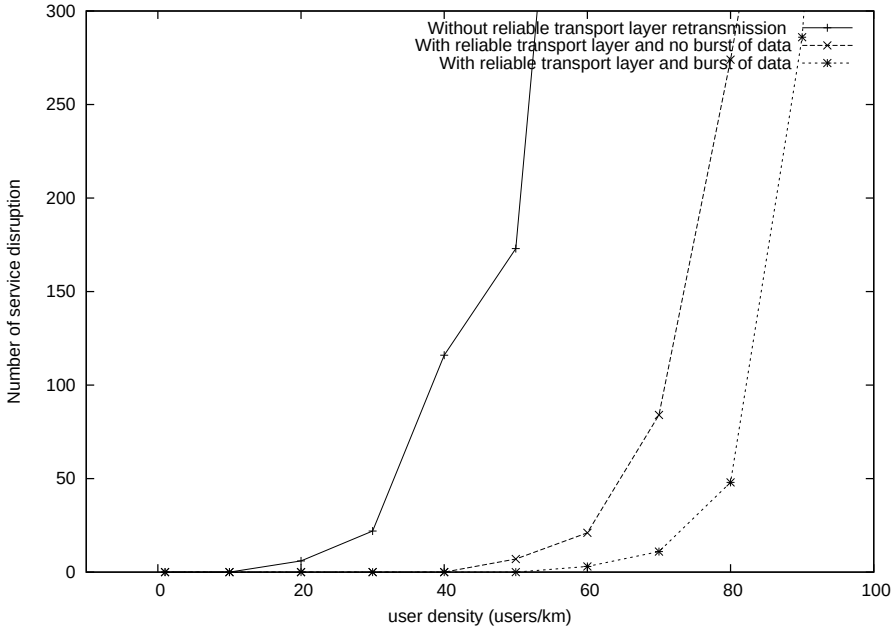

Figure 4. Evolution of service disruptions using an anchor point

Finally this new logically discontinuous vision of the cellular networks makes our architecture model (one AC / anchor point inserted between terminals and data servers) fit to classical 3rd generation networks, which are continuously covered but subject to important bandwidth fluctuations, as well to Pico-cells networks.

5.3.3. Design of a data storage model

As we have said before, several studies are done to provide efficient services in the down-link (from servers to terminals). We now strong believe that many interesting services may exist in the up-link. The idea is to benefit from our network architecture, mainly, the existence of caches in ACs and terminals to provide new (up-link) services. The latter are possible thanks to the advances in the mobile terminals. Mobile phones with cameras are in widespread use. PDAs and laptops have WiFi interfaces for mobile computing. A wireless device can be used for sending/receiving photos, videos, emails. Many problems appear because of the mobile environment and technological limitations. For example, the radio bandwidth, network connectivity, storage capacity. Users want to move in different areas without worrying about their applications. For instance, a tourist in Paris takes photos of the "Eiffel tower" with his digital camera, and then he moves to take photos of the historic "Versailles palace", but he has not got any free memory storage left. The tourist wants to enjoy the view and does not like to search for a high bandwidth connection or an extra memory storage. Similar problems appear with many other applications like movies, video clips and shared memory storage.

In this context, we propose a framework that represents a new distributed storage model that implicitly exchanges data with the closest AC when a high bandwidth connection is available. Terminals store data in their local caches, modify them while disconnected, and synchronize them with the closest AC when a high bandwidth connection is detected. For doing so, we propose a communication model adequate with the proposed architecture. It is based on the client server model taking into account the distributed and hierarchical architecture. ACs are used to temporary store incoming data from terminals and servers. The delay time the data stay in an AC depends on a class of service **(CoS)** associated with data. For example, a tourist may need to

send photos to his family immediately or as soon as possible. Another tourist may only need a shared memory for his album without sending the photos at all [14].

All the system mechanisms needed for the storage model have been specified by Mazen Tlais (last year PhD). We have also identified some applications that can exploit the up-link and the CoS principle. At present our goal is (1) to integrate these mechanisms with the scheduling policy already implemented in ACs, and (2) to demonstrate the pertinence of this approach by experimenting identified applications.

5.4. Java Operating Systems

Participants : Fabien Allard, Michel Banâtre [contact], Arnaud Guiton, Cédric Mosch, Jean-Paul Routeau, Claude Vittoria, Pierre Duquesne, Ciarán Bryce.

Our work on Java operating systems has been going on for four years now. At the end of 2004, we had completed the design of a Java operating system for the underlying Java processor and proposed the idea of *native objects* to encapsulate hardware devices. We had started to implement the system over a Linux platform and, in this context, implemented the TCP/IP stack in the Java language. In addition to our work with Texas Instruments, our group has also been collaborating on a Java Specification Request for resource management, and on a security model for Java multi-tasking environments with researchers from Sun Microsystems.

5.4.1. Pure Java Platforms

Last year (2005), we concentrated on the specification of our Java abstract machine around the bytecode processor. In particular, we formalised the integration of native objects and interrupt handling into the machine. This leads to the layered operating system model that is illustrated in Figure 3. In this model, the methods of native objects are coded as a special bytecode instructions that are micro-coded using the instruction set (C-ISA) supported by the associated hardware device.

There has been a considerable amount of implementation work on the project. We started coding a software implementation of the bytecode processor based on the K virtual machine (KVM); this yielded what we name the KVM BYTECODE PROCESSOR (KBP). Native objects were developed for a frame-buffer based screen and a keyboard. Device drivers were written in the Java programming language for a UART serial bus, the screen and keyboard devices, and for an infrared (IRDA) device. At the Java service layer (c.f., Figure 3), we developed a thread scheduler in Java for the CLDC variant of the Java (J2ME) environment and at the operating system services layer, we developed a protocol stack for the GPRS mobile phone protocol and integrated this with the IP stack that we developed last year.

Two PhD students are currently working on the our Java operating system project. *Cédric Motsch* is integrating multi-tasking (for running multiple applications) over the operating system and wrote a cross-application scheduler. The main innovation in the platform is the introduction of application domains. A domain can encapsulate an application and its JDK and can be isolated from other domains. Restricted sharing between domains is nonetheless possible. *Arnaud Guiton* is currently studying memory management for the Java platform.

One of the new topics that we began studying this year is software patching. The code base of systems and applications are dynamic: they continue to change even after delivery of the software to clients. These changes are implemented through the application of *patches*. Though their primary necessity is to fix security holes and discovered bugs, patches are even used to add new functionality. Patching requires a formal framework that ensures that modifications can be safely made to running applications.

5.4.2. Resource Management

Independently of the Texas Instruments collaboration, the group participated in a Java Specification Request for a resource management interface for the Java platform. Resources include CPU, heap memory, network bandwidth; a systems programmer can define further abstract resources, e.g., database connections, servlet invocations, etc. Monitoring resource consumption and imposing limits on consumption are key requirements for any platform. Until now, no such mechanism existed for the Java platform. The JSR was in progress for

nearly two years, and its output is a Java API and reference implementation. The API specification has now entered a public review stage. In the context of JSR 284, we collaborated with representatives from Google, IBM, BEA Systems, Intel, Nokia, Siemens, Motorola and EPFL-Switzerland. This collaboration led to the final draft of the specification, published in December 2006.

6. Contracts and Grants with Industry

6.1. National contracts

6.1.1. Aéroport de Paris

- Title: Conception et mise au point de pilotes de services aéroportuaires reposant sur les technologies de l'informatique diffuse dveloppées par l'INRIA- UR Rennes.
- Partner: *Aéroport de Paris*
- Starting: 01/11/2006, ending : 15/02/2007

6.1.2. TV mobile <<sans limite>> (TVMSL)

- Partner : Alcatel Mobile Broadcast, Alcatel Alenia Space Alcatel, DiBcom, Sagem Communication, TeamCast, Radio Frequency Systems, UDcast, CEA-Leti, Inria-Rennes, CNRS-L2S
- Starting: 01/11/206, ending : 30/04/2009

The objective of this contract is to design and evaluate inovative caching mechanisms to distribute H.264 flows in future hybrid DVB-H+ infrastructure (satellite + terrestrial).

6.1.3. Texas Instruments

- Number: 198C2730031303202
- Title: Real time Java Distributed Processing Environment
- Related Research activity: see section 5.4
- Partner: *Texas Instruments*
- Founding: *Texas Instruments*
- Starting: 01/10/1998, ending : 30/09/2001
- Extension starting: 01/10/2001, ending : 30/09/2003
- Extension starting: 01/10/2003, ending : 30/09/2006

The objective of that contract is to design and build a Java runtime suitable for embedded platforms on Texas Instruments hardware architectures.

6.1.4. Alcatel

- Number: 101C078
- Title: Advanced service delivery for 4G discontinuous networks
- Related Research activity: see section 5.3
- Partner: *Alcatel*
- Founding: *Alcatel*
- Starting: 01/09/2002, ending : 30/10/2004
- Extension starting: 01/11/2004, ending : 31/10/2007

The objective of that contract is to design, build and experiment an efficient service data delivery in heterogeneous mobile networks. The targeted solution has to be distributed between the network components and the mobile terminal.

6.1.5. JCDecaux

- Number: Inria 401
- Title: Tuple-ware,
- Related research activity: see section 5.2
- Partner: *JCDecaux*
- Founding: *JCdecaux*
- Starting: 01/11/2004, ending 31/05/2005

The goal of this contract was to implement a pilot (named Tuple-ware), in order to help the JCDecaux company to evaluate the potential of ambient computing services based on ACES technologies in the area of street furniture and billboard.

7. Other Grants and Activities

7.1. European actions

7.1.1. European Project: Roboswarm

- Title: Robot Swarms
- Proposal/contract no: 045255
- Tallinna Tehnikalikool- Estonie, ELIKO Tehnoloogia Arenduskeskus- Estonie, Institut National de Recherche en Informatique et en Automatique- UR Rennes, TEKNILLINEN KORKEAKOULU- Finlande, OULUN YLIOPISTO- Finlande, FUNDACION FATRONIK- Espagne, KTH - Kungliga Tekniska Hgskolan Royal Institute of Technology- Sude, IDMIND-ENGENHARIA DE SISTEMAS, LDA -Portugal, Universita degli Studi di Genova -Italie
- Starting: November 2006, ending: April 2009

Roboswarm is an EU project that started in November 2006 in which ACES is a participant. The goal of the project is to develop an open knowledge environment for self-configurable, low-cost and robust robot swarms usable in everyday applications. Advances in the state-of-the art of networked robotics are proposed through introduction of a local and global knowledge base for ad hoc communication within a low-cost swarm of autonomous robots operating in the surrounding smart IT infrastructure. The ACES group was invited into this consortium due to its experience with ad hoc (ambient) network environments.

7.1.2. Coordinated Action: Embedded WiSeNts

- Title: Cooperating Embedded Systems for Exploration and Control featuring Wireless Sensor Networks
- Partners: Technische Universität Berlin (Germany), University of Cambridge (UK), University of Copenhagen (Denmark), Swedish Insitute of Computer Science (Sweden), University Twente (Netherlands), Yeditepe University (Turkey), Consorzio Interuniversitario Nazionale per l'Informatica (Italy), University of Padua (Italy), Swiss Federal Institute of Technology Zurich (Switzerland), Asociacion de Investigacion y CooperacionIndustrial de Andalucoa (Spain), Institut National de Recherche en Informatique et en Automatique (INRIA), Universität Stuttgart (Germany)
- Starting: September 2004, ending: December 2006

Embedded WiSeNts aims to increase the awareness and to find out a vision as well as a researh roadmap towards wireless sensors networks of cooperating embedded systems, within the academic community and, most importantly, within the manufacturers of proper technologies as well as potential users community.

7.1.3. NoE Resist

- Title: Resilence and Survability for IST
- Head: LAAS
- Starting: beginning of 2006

The NoE ReSIST (Resilence and Survability for IST) will focus on the following four objectives in addressing the scalability of dependability and security via resilience:

- Integration of teams of researchers so that the fundamental topics concerning scalably resilient ubiquitous systems are addressed by a critical mass of co-operative, multi-disciplinary research.
- Identification, in an international context, of the key research directions induced on the supporting ubiquitous systems by the requirement for trust and confidence in AmI.
- Production of significant research results that pave the way for scalably resilient ubiquitous systems.
- Promotion and propagation of a resilience culture in university curricula and in engineering best practices.

7.2. French initiative for research in security and informatics

7.2.1. ACI: Mosaic

- Title: Mobile System Availability Integrity and Confidentiality
- Partners: Institut Eurecom, Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA), Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS)
- Starting: September 2004, to August 2007

The MoSAIC project is studying new fault tolerance and security mechanisms for mobile wireless devices in ambiant intelligence applications. We focus on sparse self-organized networks, using mostly one-hop wireless communication.

8. Dissemination

8.1. Animation of the scientific community

8.1.1. Program committees

- PC member of the International Workshop on Privacy and Security in Agent-based Collaborative Environments, PSACE 2006. (C. Bryce)
- PC member of 2007 ACM Symposium on Applied Computing, Track Security. (C. Bryce)
- PC member for the 5th Conference on Security and Network Architectures (SAR 2006), Mont de Marsan, France, 2006. (C. Bryce)
- PC member of Algotel 2007, *9ème rencontres francophones sur les aspects algorithmiques de télécommunications*. (F. Weis)
- PC member for the Fourth International Conference on Cooperative Internet Computing (CIC2006) Hong Kong, China, 25-27 Oct 2006. (M. Banâtre)
- PC member for the 3rd European Workshop on Wireless Sensor Networks (EWSN), ETH Zurich, Switzerland, February 13 - 15, 2006. (M. Banâtre)
- PC member for the 18th International Symposium on Personal, Indoor and Mobile Radio Communications, Athens, 3-7 September 2007. (M. Banâtre)

- PC member for the 5th IFIP Workshop on Software Technologies for Future Embedded & Ubiquitous Systems (SEUS 2007, part of ISORC 2007). (P. Couderc)
- PC member for Self adaptability and self management of context-aware systems, (part of ICAS 2006). (P. Couderc)

8.1.2. Organizing and reviewing activities

Ciarán Bryce is a member of the expert scientific committee for the ANR (*Agence national de recherche*) Global Security program.

Michel Banâtre is member of the scientific committee of the "encyclopedie des systèmes d'information" (Vuibert) and scientific leader of the section related to Architectures and Operating Systems (end of 2005).

8.2. National and international working groups

Participation to the GdR I3 (Information - Interaction - Intelligence), group Mobility and Ubiquitous Computing (F. Weis)

8.3. Teaching activities

- Ifsic
  - Responsability of the optional lecture on Operating Systems in masters in Computer Science (M. Banâtre, P. Couderc, F. Weis),
  - Responsibility of the lecture on Ambient Computing and Distributed Operating Systems in "Diic 3 ARC" (final year of masters) (M. Banâtre, P. Couderc and F. Weis),
- Ecole des Mines de Nantes
  - Responsability of the lecture on distributed systems (final year of masters) computer science department (M. Banâtre),
- INSA of Rennes
  - Responsibility of the lecture on Ambient Computing and Distributed Operating Systems (final year of masters) computer science department (M. Banâtre).
  - 6 hours of lectures on Computer Security to final year engineering students (C. Bryce).
- University of Rennes I, UFR "Structure et Propriétés de la Matière", Lecture in DESS "DRI" (French equivalent of Master's Degree) on home networking (F. Weis).
- ENST Bretagne, Lecture on Wireless LANs (final years of masters) (F. Weis).
- ENSEIRB (Bordeaux), Conference on Mobile communications and ambient computing, final year of masters, October 2006 (M. Banâtre).

F. Weis participates to the computer science *commissions de spécialistes* of universities and schools: Université de Lille 1.

8.4. Internship supervision

We have supervised the following internships in 2006:

- Julien Bigot (Engineer student, INSA Rennes)
- Azza Jedidi (Engineer student, ENSEIRB Bordeaux)

8.5. Patents

- *Dispositif de communication local à selection sur base contextuelle*. January 2006 (Demande PCT/FR0601031). Patent improving the PCT/FR01/01409 in the context of the RFID technology.

8.6. Industrial transfers

The strong point of the group is its *research methodology*. We have consistently sought industrial collaboration for our research in order to validate our assumptions and/or to ensure that solutions can be used in a real world context. Thus, our work on ambient computing models has led to the sale of several patents, including a major technology transfer to JCDecaux – the major world-wide billboard provider. Our work on continuous service provision has been done in collaboration with Alcatel, and our work on Java operating systems has been conducted in collaboration with Texas Instruments, Rennes.

The group oversaw the creation of UbiSphere in January 2006 based on our ambient computing technology. The company is currently an INRIA incubator, and will remain so until the end of 2007.

From November 2004 to May 2005, the group was contracted by JCDecaux to implement a pilot (named Tuple-ware), in order to help the JCDecaux company to evaluate the potential of ambient computing services based on ACES technologies in the area of street furniture and billboard. This collaboration led to JCDecaux buying our patent and software in February 2006.

From November 2006 to February 2007, the group has been commissioned by ADP (**Aéroports de Paris (ADP)**) to design and implement a pilot of an ambient system, based on the group's technology, for use in airport environments.

9. Bibliography

Year Publications

Books and Monographs

[1] M. BANÂTRE, C. BRYCE, P. COUDERC, F. WEIS. *Informatique diffuse : des concepts à la réalité*, to appear, Hermes Publishing, 2007.

[2] M. BANÂTRE, A. OLLERO. *Cooperating Embedded Systems and Wireless Sensor Networks*, to appear, Hermes Publishing, 2007.

[3] C. BRYCE. *Les systèmes d'exploitation*, Vuibert Encyclopedia, 2006.

[4] P. COUDERC, M. BANÂTRE. *Embedded WiSeNts Research Roadmap*, co-authors with the Embedded Wisents consortium, Logos Verlag Berlin, 2007.

Doctoral dissertations and Habilitation theses

[5] C. BONAN. *Délivrance continue de données dans un réseau sans fil à couverture discontinue*, Ph. D. Thesis, Université de Rennes I, december 2006.

[6] J. PAUTY. *Rôle de la géométrie en informatique diffuse : programmation des applications et navigation contextuelle*, Ph. D. Thesis, Université de Rennes I, February 2006.

Articles in refereed journals and book chapters

[7] M. TLAIS, F. WEIS. *Distributed communication model in hierarchical infostations systems*, in "Concurrency and Computation: Practice and Experience", To appear, 2007.

Publications in Conferences and Workshops

[8] X. LEBOURDON, P. COUDERC, M. BANÂTRE. *Spontaneous Hotspots: Sharing Context-Dependant Resources Using Bluetooth*, in "Proc. of International Conference on Autonomic and Autonomous Systems (ICAS 2006), Santa Clara, CA, USA", July 2006.

[9] D. MARTIN-GUILLEREZ, M. BANÂTRE, P. COUDERC. *Increasing Data Resilience of Mobile Devices with a Collaborative Backup*, in "Proc. of International Conference on Dependable Systems and Networks (DSN'06), Philadelphia, PA, USA", June 2006.

[10] M. TLAIS, F. WEIS, C. BONAN. *Mobility prediction in 4G D-cov Networks*, in "Proc. of the 1st IEEE International Workshop on Peformance Analysis and Enhancement of Wireless Networks (PAEMN'06), Vienna, Austria", April 2006.

[11] M. TLAIS, F. WEIS. *Centralized Mobility Prediction Support in a Hierarchical Architecture*, in "Proc. of the International Conference on Wireless and Mobile Communications (ICWMC'06), Bucharest, Romania", August 2006.

Internal Reports

[12] A. JEDIDI, F. WEIS. *Caching and scheduling mechanisms for H.264 video flow delivery over discontinuous coverage wireless networks*, To appear, currently under NDA screening by Alcatel, Technical report, IRISA, January 2007.

[13] A. LUU, F. WEIS, R. MENARD. *Improving wireless network capacity by introducing logical discontinuous coverage*, To be published in 2007, currently under NDA screening by Alcatel, Technical report, IRISA, January 2007.

[14] M. TLAIS, F. WEIS. *Data storage model in infostation systems*, To appear, currently under NDA screening by Alcatel, Technical report, IRISA, January 2007.

References in notes

[15] (INTEL CORP.). *Introduction to the iAPX 432 Architecture*, Intel Corporation, Santa Clara, CA, 1981.

[16] B. ALPERN, C. R. ATTANASIO, J. J. BARTON, A. COCCHI, S. F. HUMMEL, D. LIEBER, T. NGO, M. MERGEN, J. C. SHEPHERD, S. SMITH. *Implementing Jalapeño in Java*, in "Proceedings of the Conference on Object-Oriented Programming, Systems, Languages, and Applications", 1999, p. 314–324.

[17] K. ARNOLD, J. GOSLING. *The Java Programming Language*, The Java Series, Addison-Wesley, 1996.

[18] BURROUGHS. *Burroughs B6700 Information Processing Systems Reference Manual*, 1972.

[19] D. ESTRIN, R. GOVINDAN, J. HEIDEMANN. *Embedding the Internet*, in "Communications of the ACM", vol. 43, n^{o} 5, May 2000, p. 39–41.

[20] E. FREEMAN, S. HUPFER, K. ARNOLD. *JavaSpaces Principles, Patterns, and Practice*, Addison-Wesley, Reading, MA, USA, 1999.

[21] D. GELERNTER. *Generative Communication in Linda*, in "TOPLAS", vol. 7, n^{o} 1, jan 1985.

[22] K. HAMEED. *The Application of Mobile Computing and Technology to Health Care Services*, in "Telematics and Informatics", vol. 20, n^{o} 2, 2003, p. 99–106, http://dx.doi.org/10.1016/S0736-5853(02)00018-7.

[23] B. NOBLE, M. SATYANARAYANAN, J. TILTON, J. FLINN, K. WALKER. *Agile application-aware adaptation for mobility*, in "Proceedings of the 16th Symposium on Operating Systems Principles", 1997.

[24] C. WAWERSICH, J. KLEINDER, M. FELSER, M. GOLM. *The JX Operating System*, April 22 2002, http://citeseer.ist.psu.edu/561091.html.

[25] M. WEISER. *Some Computer Science Issues in Ubiquitous Computing*, in "Communication of the ACM", vol. (7)36, 1993, p. 75-83.

[26] M. WOODGER. *An introduction to ALGOL 60*, in "The Computer Journal", vol. 3, July 1960, p. 67–75.

[27] W. WULF, E. COHEN, W. CORWIN, A. JONES, R. LEVIN, F. POLLACK. *HYDRA: The Kernel of a Multiprocessor Operating System*, in "Communications of the ACM, CACM", vol. 17, n^{o} 6, June 1974, p. 337–345.

[28] P. WYCKOFF, S. MCLAUGHRY, T. LEHMAN, D. FORD. *T Spaces*, in "IBM Systems Journal", vol. 37, n^{o} 3, 1998, p. 454–474.