INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

# *Project-Team Jacquard*

# *Weaving of Software Components*

*Futurs*

THEME COM

# Table of contents

# 1. Team

*Jacquard is a joint project between INRIA, CNRS and Université des Sciences et Technologies de Lille (USTL), via the Computer Science Laboratory of Lille : LIFL (UMR 8022).*

**Team Leader**
Jean-Marc Geib [Professor, USTL]

**Administrative Assistant**
Axelle Magnier [Assistant project INRIA]

**Staff member INRIA**
Philippe Merle [Research associate]
Renaud Pawlak [Research associate]
Lionel Seinturier [Research associate (secondment INRIA)]

**Staff member LIFL**
Olivier Caron [Associate Professor Polytech'Lille]
Bernard Carré [Associate Professor Polytech'Lille]
Laurence Duchien [Professor USTL]
Anne-Françoise Le Meur [Associate Professor USTL]
Raphaël Marvie [Associate Professor USTL]
Gilles Vanwormhoudt [Associate Professor Telecom Lille I]

**Ph. D. student**
Olivier Barais [MESR grant until December 1st 2005]
Dolorès Diaz [NorSys CIFRE grant]
Jérémy Dubus [MESR grant since October 1st 2005]
Frédéric Loiret [CEA grant]
Naouel Moha [University of Montreal, since November 1st 2005]
Alexis Muller [Assistant Professor, ATER]
Carlos Francisco Noguera Garcia [AOSD grant since September 1st 2005]
Nicolas Pessemier [France Télécom grant]
Romain Rouvoy [INRIA-Région grant]

**Post-doctoral fellow**
Maja D'Hondt [Post-doctoral fellow - ERCIM Grant, since June 1st 2005]
Patricia Serrano-Alvadarro [Post-doctoral fellow, 2004-2005 until August 31]

**Invited Scientist**
Wolfgang De Meuter [VUB, June-September 2005]
Julia Lawall [DIKU, May 2005]

**Project technical staff**
Christophe Contreras [Project staff - ITEA OSMOSE until June 30th 2005 - ITEA S4ALL, December 2005]
Cédric Dumoulin [Project staff - ITEA OSMOSE until June 30th 2005]
Areski Flissi [Technical staff CNRS]
Jérôme Moroy [Project staff - ITEA OSMOSE until August 31st 2005]
Nicolas Petitprez [Project staff INRIA - since September 1st 2005]
Missi Tran [Project staff INRIA - July -December 2005]

# 2. Overall Objectives

## 2.1. Overall Objectives

**Keywords:** *Aspect-Oriented Programming (AOP), Component Weaving, Component Models, Component-Based Adaptive Middleware (CBAM), Integrated Tools for Production and Exploitation of Software Compo-*

*nents, Model-Driven Software Engineering (MDSE), Run-time Containers, Separation of Concerns (SoC), Software Architecture.*

### 2.1.1. J.M. Jacquard and Weaving Machines

One of the first historical steps towards programming appeared in 1725 on a weaving machine. The French "Lyonnais" Basile Bouchon first gives instructions to a weaving machine using a perforated paper. His assistant Mr Falcon replaces the fragile paper by more robust perforated cards. After that, Mr Vancanson replaces the cards by a metallic cylinder and a complex hydraulic system, which gives the machine a cyclic flow of instructions a program!

But History keeps in mind Joseph-Marie Jacquard who creates and commercialises the first automatic weaving machine during the beginning of 19th century. The precision of the machine allows Joseph-Marie Jacquard to design a program that weaves his own face on a fabric. Joseph-Marie Jacquard innovations have greatly contribute to first steps of computer science with the perforated cards to support programs. The idea of independent programs for a programmatic machine was born!

### 2.1.2. Context and Goals

The Jacquard project focuses on the problem of designing complex distributed applications, i.e., those composed of numerous cooperative and distributed software components, which are constrained by various requirements, such as persistency, security and fault tolerance. We want to investigate the ability of software engineers to produce new component-oriented platforms and new methodological and technical approaches to design and exploit these applications. In particular, we explore the use of component models, separation of concerns and weaving in the different phases of an application's life cycle (i.e., modelling, design, assembling, deployment, and execution). Our goal is to produce fully functional platforms and tools. Finally, we are members of standardization organizations (OMG) and the open source software world (ObjectWeb).

# 3. Scientific Foundations

## 3.1. Weaving of Software Components

One of the challenges for software components is to build new models and platforms to allow large scale interoperability of components for designing complex distributed applications. Actually, some models exist: Enterprise Java Beans by Sun Microsystems, .Net by Microsoft and the CORBA Component Model in the CORBA3 OMG standard [96]. These models and platforms are clearly not satisfactory because of the lack of functional completeness and interoperability. Moreover, the industrial propositions only deal with a lot of technical problems to capture the component software notion, but mainly forget the needs to manipulate the models of components and applications independently of the technical aspects. This point has been recently tackled by OMG with its Model Driven Architecture (MDA) initiative [94], [101]. We agree that these points (Component Models, Component oriented Platforms and Model Driven Engineering) lead to new research problems in the goal to produce a better integrated product line from analysis to exploitation for component based applications.

Jacquard members have a great research experience in two computer science domains related with the goal of the project: Jean-Marc Geib, Philippe Merle and Raphaël Marvie have some important contributions in the Distributed Object based Platforms area [71], Laurence Duchien, Bernard Carré and Olivier Caron on specifications and use of separation of concerns for complex applications. For example, we can quote the contributions to the OMG standardization work with the CorbaScript language [89] (proposed to the Scripting Language for CORBA RFP, and accepted as the CORBA Scripting Language chapter of CORBA3 [80]) and with the CCM (CORBA Component Model) chapter for which we lead the response group and the revision task force. Other examples are the JAC (Java Aspect Component) platform, one of the leading platforms for dynamic weaving of aspects [97], [99], [100], and the View Approach for structuring the design of information systems [110].

We aim to associate these experiences to design and produce an ambitious new platform for component based complex applications with new methodological and technical traits for structuring the large set of hardly related problems in supporting theses applications. Models, platforms and applications have to benefit from new open middleware using separation of concerns and weaving. Our contributions want to understand how a better structure of models and platforms can give better software for complex applications.

For the next four years, the project's goals are:

- to produce a full platform for the CCM model. This platform, called OpenCCM, has to contribute to the OMG standardization work. Moreover it will provide new adaptable containers allowing the weaving of system aspects, dynamically following the application requirements. It will also provide an integrated environment to manipulate, deploy and exploit assemblies of components.
- to define a complete design and technical environment for assembling of components and aspect, via a dedicated modelling tool for composition and a dynamic aspects oriented platform that will be a next step of our JAC platform.

## 3.2. OpenCCM

This part of the project deals with the design and the production of new tools for component based platforms. This work was initiated in the Computer Science Laboratory of Lille (LIFL) and is now one of the projects of the ObjectWeb Consortium [73] under the name OpenCCM. Our goal is a full platform for the OMG's CORBA Component Model (CCM). We want to fully capture all the aspects of this norm and contribute to it. Our ambition is to produce the first referenced CCM platform in an open source format. Actually OpenCCM is already a LGPL software accessible at http://openccm.objectweb.org. Beyond this production we aim to investigate three points as research topics: open the platform to allow extensibility and adaptability, open the run-time containers to weave non-functional aspects, and give the capability to freely assemble components in an open environment. These three points are detailed in the next sections. This work is related to other works on open middleware: the Fractal model [66] for component middleware (ObjectWeb, INRIA Sardes project, France Telecom), reflexive middleware approaches (Dynamic TAO [78], Flexinet [72], OpenCorba [82], OpenORB [104]), adaptable middleware approaches (ARCAD RNTL project [83]), virtual machines (VVM) and QoS driven Middleware [74].

### *3.2.1. Open Middleware for the CCM*

The OpenCCM project proposes an open framework to produce and exploit CORBA Components. One can specify such a component in the new OMG IDL3 language, which is an extension of the old CORBA IDL2 language. The framework can produce IDL2 schema from IDL3 descriptions, and the associated stubs for various programming languages (Java, C++, IDLscript, ...) [87]. The framework is itself composed of reusable components around an IDL3 global repository. This architecture is open and extensible. The components are written in the Java language and are also CORBA components, so that they can be assembled to create several configurations. So the platform can be instantiated in several way for middleware such as ORBacus, OpenORB or Borland Enterprise Server.

Current work plans to complete the framework with the Component Implementation Definition Language, the Persistent State Definition Language, and the JORM framework. This will allow the platform to automatically generate containers with persistency capabilities. We work also on the assembly and packaging tools using the XML descriptors of CCM, and we also work on the transformation tools towards C++.

### *3.2.2. Open Containers*

A major goal of component based platforms is to be able to separate functional aspects (ideally programmed by an expert of the tackled domain) from the non-functional aspects (ideally programmed by an expert of the computer system techniques). This separation can be implemented by a technical separation between the components (functional aspects) and the containers (non-functional aspects). A container hosts components, so that the components inherit the non-functional aspects of the container.

Actually containers (such as the EJB or CCM containers) can only contain a limited set of non-functional aspects (activation/termination , communications and events, security, transactions and persistency). These containers are not extensible neither statically nor dynamically. So they cannot respond to specific needs such as fault tolerance, replication, load balancing, real- time or monitoring.

We plan to design these open containers. We investigate a generic model for containers and the weaving mechanisms which will allow an application to specify particular needs. So an application will be able to reclaim the deployment of well-fitted containers. We work on a specific API to develop non-functional aspects for our containers. In a first step we have to specify a large number of non-functional aspects to find the way to compose them. Non-functional aspects can be seen as interceptors, so we work on composition of interceptors to produce containers. In a second step we will investigate the possibility to dynamically manipulate the containers to change the configuration of non-functional aspects.

### *3.2.3. Open Environment*

An open environment for component-based applications has to deal with several problems. For instance, we have to allow assemblies and deployment on demand. In this part we plan three goals: a virtual machine for programming distributed deployments, a trader of components to realize assemblies from 'off the shelves' components, a repository to manipulate and drive assemblies of components.

Current middleware propose fixed deployment strategies which are not adaptable to specific needs. These deployment tools are mainly 'black boxes' and ad-hoc in a particular environment. In the CCM context we can exploit the XML based OSD language which is used to describe assemblies. This is a good basis to describe deployments. But the CCM does not define an API to control the deployment and the associated tools have not be realized for today in an open manner. Actually we work on a set of operations to deploy OSD assemblies. We investigate several useful properties (such as optimised deployment, parallel deployment, fault tolerant deployment, transactional deployment) implemented by these operations. This will lead to an open API for adaptable deployment strategies [86], [85]. We plan to use IDLscript to specify the strategies.

Assemblies can be constructed on demand with 'Components Off The Shelves'. We work on this point with our TORBA environment [81]. Within TORBA we can instantiate components for trading from trading contracts (specified in our TDL - Trading Description Language). This is the basis for an open infrastructure for components brokering that we plan to investigate here.

In an open framework for components we have to manipulate assemblies in all the phases of the design work and also during execution. Assemblies have to be manipulated by various users, each with his/her own concerns (e.g., assemble, deploy, distribute, non-functional aspects set-up, monitoring). We plan to construct a global repository for all these activities. Moreover this repository has to be opened for new activities. In this way we want to define an environment which allows for the defintion, at a meta level, of the different concerns that we want to exist on the repository [84]. Then the environment will be able to automatic generate a new view on the repository to capture the specified activity [79].

## 3.3. Aspects Oriented Design of Dynamic Components Assemblies

The behaviour of a complex application in an open environment is difficult to specify and to implement because it has to evolve in accordance with the context. Changes can occur in an asynchronous manner and the behaviour has to be adapted without human actions and without stopping the application. A language to specify an assembly of components has to capture these dynamic aspects. A platform which supports the assembly at run-time also has to be able to respond to the needed changes. In this part of the project we plan to investigate three directions.

The first one deals with the study of separation of concerns from the first steps of analysis to the implementation. The goal is to trace the evolution of different concerns in these various stages. The second one is related to the dynamic features of the Architecture Description Languages (ADL) [75], [88]. The last one focuses on Aspect Oriented Programming [77] [98] [69] in which one can capture a specific concern of a behaviour.

Finally, this project part enhances specifications of component assemblies in the goal of designing adaptable applications. We introduce integration contracts for specifying the impact of components on the application and its context [55], [65]. Our approach is based on AOP to specify connection and integration schemas.

### *3.3.1. Early Aspects*

Business applications are faced with two main challenges. On one hand, they are mostly developed using an iterative process where business functionalities are added to the core application as the project requirements evolve [48]. On the other hand, the non-functional requirements (in terms of security, remote communication and transaction, data persistence, etc.) are also high and need to be incorporated as seamlessly as possible. Both the component-based and the aspect-oriented approaches separatively provide directions for these challenges. However it exists no integrated software processes that takes both into account. The goal of this work is thus to propose such a process and some tools to support it starting at the early stages of analysis and to provide features to trace their evolution from user requirements until deployment and run-time. This work is done in the context of Dolores Diaz's PhD thesis [23].

### *3.3.2. Aspects at Design-time*

Software architects and designers use reasoning frameworks to iteratively integrate functional and non-functional concerns into their projects, and to adapt them to unforseen functional or non-functional requirements. To assist them, analysis methods support modelling and verification tools from functional to technical architecture. Their main advantages are to model large-scaled distributed systems that require interoperability between system parts and the separation of concerns between business functionality and communication mechanisms.

However, no standard and universal definition of the software architecture has been accepted by all the community [88], [49], [50], [57]. Various points of view on different studies bring to several approaches. These approaches focus on only one or two concerns such as component interfaces specification, behavioural analysis or software reconfiguration. So we argue that, in order to increase benefits of software architecture approaches, one may need to use an architecture centric approach with a global reasoning: from software architecture design to software architecture management to software architecture building, deployment and refinement. However, these different concerns of a software architecture definition must be kept consistent.

Our first goal is to propose enhancements of a component model for specifying dynamic evolution of an software architecture. It concerns three points of view: structural, functional and behavioural points of view [91], [53], [51]. We use Model Driven Architecture approach with Context Independent Model and Context Specific Model. Our second goal is to introduce non-functional aspects and connexions between components and containers in languages for software architectures. We extend contracts between components to contracts between components and non-functional components [52].

### *3.3.3. Aspects at Run-time*

In distributed environments, applications run in an open context. They use networks and their associated services for which quality of service is not always guaranteed and may change quickly. In these environments, several concerns must be considered, including fault tolerance, data consistency, remote version update, runtime maintenance, dynamic lookup, scalability, lack of rate. Addressing these issues may require dynamic and fast reconfiguration of distributed applications [107], [90], [105].

We have defined the Java Aspect Components (JAC) framework for building aspect-oriented distributed applications in Java [97] [100], [99]. Unlike other languages such as AspectJ, JAC allows dynamic weaving of aspects (aspects can be weaved or unweaved at run-time) and proposes a modular solution to specify the composition of aspects. We defined an aspect-oriented programming model and the architectural details of the framework implementation. The framework enables extension of application semantics for handling well-separated concerns. This is achieved with a software entity called an aspect component (AC). ACs provide distributed pointcuts, dynamic wrappers and metamodel annotations. Distributed pointcuts are a key feature of our framework. They enable the definition of crosscutting structures that do not need to be located on a single

host. ACs are dynamic. They can be added, removed, and controlled at runtime. This enables our framework to be used in highly dynamic environments where software adaptation is needed.

## 3.4. Functional Aspects for Components Applications and M.D.E.

Software engineering aims at being rationalized always more and begins to reach levels of productivity and reuse that come near to other fields such as mechanics or electronics. The challenge is to facilitate the production of software that are more and more complex, robust, maintainable and evolutive. Component oriented design is a recent step towards that productivity through re-usability. It allows the composition of "off the shelf" software entities, while preserving good properties on the software. The composition mechanisms are mainly used for construction and deployment phases, but the modelling phases are often not addressed by these ideas around composition.

After being considered only as documentation elements for a long time, models are gaining more and more importance in the software development lifecycle, as full software artefacts. The UML [95] standard contributes a lot to this mutation, with the identification and the structuration of models space dimensions and constructs. Models can nowadays be explicitly manipulated through metamodeling techniques, dedicated tools or processes such as the MDA [94] transformation chains. This is "Model Driven Engineering" [76].

The main motivation is the reduction of delays and costs by the capitalization of design efforts (models) at each stage, and the automation, as far as possible, of transitions between these stages. Thus, it would be possible to separate high level business oriented models from low level architectural and technological ones, but also to reuse these models from one application to another. Indeed, once it is clear that models are full software ingredients, we are faced with new problems such as the possibility of their reusability and composability. As a consequence, models stand more and more as good candidates for the "design for reuse" quest and specific constructs are introduced to make them generic.

We want to investigate the idea that functional decomposition of models is a way for increased re-usability. Our interest takes place in the use of functional aspects which represent the various dimensions of a tackled domain. It is related to aspect oriented structuring, and design plans like the views, SOP [64] and Catalysis [67] approaches. We start from our previous works on the application of view structuring to object-oriented programming [110], databases [68], [61] and software architectures [60], [84].

The problem of the reuse of these functional aspects arises now. Indeed, this reuse must make it possible to improve the productivity and reliability in the field of information systems design. Several approaches propose the reuse of functional aspects in various forms, such as the design of reusable frameworks [67] or in the form of UML templates [64].

Our goal is to 'disconnect' functional views from a specific domain in order to obtain functional components which will be adaptable to various contexts. This is a way for functional re-usability. Such a functional component has to capture a functional dimension with a high level of abstraction. Our idea is to introduce the notion of 'model components' parameterized by a 'required model' that produces a 'provided model'. Thus, the modelling phase can be seen as the assembly of such components by connecting provided model to required model. Note that component ports (specified by a model) can be more sophisticated than simple interfaces of objects or software components.

As a first step, we have formalized such a component model [92] and its associated design and assembly rules as an extension of the UML meta-model. We obtain adaptable model components that can be targeted to the EJB platform and the CORBA component model. We have realized an implementation of this work via an UML profile. The corresponding UML Objecteering module is available at http://www.lifl.fr/ mullera.

In [93] we compare techniques for composing and parameterizing models and keep the advantages of the later ones to specify reusable functional aspects. Model parameterization is related to templates notions, such that found in the UML scope. Applications of templates are numerous and various, with the result that its initial introduction in UML1.3 was deeply revisited and strengthened in the UML2 standard. Though its specification remains much more structural and verbal in [95]. Particularly, constraints lack a precise definition of the related template binding relationship. That is why we propose in [62] a set of OCL constraints [111] which

strengthen the notion of model templates and facilitate its exploitation in construction processes. Our model components can be expressed by UML template packages. We also identify that the UML specification needs to be extended in order to make templates parameterizable by complex models. We are defining a set of OCL constraints which formalizes this extension. We use this extension in order to define a process where package templates are composed to build a system.

A second dimension of our work is concerned with the preservation of the 'functional aspects oriented design style' from the modelling phase to the exploitation phase. This approach gives several advantages: re-usability at the modelling phase leads to re-usability at the production phase, designers can trace the design work in the exploitation of the application. So our work can be a contribution to a seamless integration of modelling tools and component based platforms such as OpenCCM or EJB.

We identify some structural patterns which allow to target functional decomposition onto component platforms. In [59], we present a composition-oriented approach grounded on the splitting of entities according to views requirements. Two original design patterns are formulated and capture the main issues of the approach. The first one is concerned with the management of the split component and its conceptual identity. The second one offers a solution for relationships among such components. These patterns improve evolution and traceability of views and can be applied to different technological platforms. An experimentation in the Fractal component model is presented in [15] which allows to envision the usage of Fractal controllers to manage split components. In [63] we focus on the reuse of functional aspects or views at the implementation level using adaptation techniques. The reuse of views is ensured by applying the adapter pattern [70]. We show how to compose the views pattern and the adaptation one. The result provides an implementation of reusable functional aspects that can be composed at the exploitation stage.

At a practical stage, all this work is gradually integrated in Case Tools (Objecteering, Eclipse Plugin), as functional aspect oriented modelling and design facilities to support model driven processes.

# 4. Application Domains

## 4.1. Application Domains

The Jacquard project addresses the large problem of designing complex distributed applications composed of numerous cooperative and distributed software components. Our application domains are numerous. First, our component models and platforms target information systems. These software need properties such as functional and technics and they must evolve. Second, component models tackle several specific domains needing adaptability of process context such as mobility or ubiquitous computing. We apply our work in transportation or communication domains, for example in MOSAIQUES project or AOSD NoE. Finally, we participate to platforms definition for grid computing.

# 5. Software

## 5.1. Apollon

**Keywords:** *Graphical Editor*, *Model Driven Software Framework*, *XML*.

**Participants:** Christophe Contreras, Philippe Merle [correspondant].

Apollon is a model driven software framework to generate Java-based graphical editors for XML documents.

According to a XML DTD given as input, the Apollon's code generator produces a set of Java Data classes and Java Swing components implementing graphical editors for XML documents. The Java Data classes are a strongly typed reification of the XML DTD: each XML DTD element is reified as a Java class, XML DTD children and attributes are reified as getter and setter Java methods. The Java Swing components implement

the graphical representation of the Java Data classes. The graphical representation of any XML element and attribute could be customized at the generation time according to users' graphical requirements.

The Apollon's code generator is built as an extension of the open source Zeus software. The Apollon's runtime is based on the Fractal Explorer software framework described below. Apollon is already used in OpenCCM to automatically generate graphical editors for the XML DTDs defined in the OMG's CORBA Components Specification.

Apollon is a LGPL open source software available at http://forge.objectweb.org/projects/apollon.

## 5.2. Fractal Explorer

**Keywords:** *Fractal Component-Based Software Framework, Graphical User Interface, Management Console*.

**Participants:** Philippe Merle [correspondant], Jérôme Moroy.

Fractal Explorer is a generic Fractal component-based software framework to build Java-based graphical explorer and management consoles.

Fractal Explorer is composed of the Explorer Description Language, the plug-in programming interface, and the Fractal component-based explorer framework. The Explorer Description Language (a XML DTD) allows users to describe at a high level the configuration of graphical explorer consoles to build, i.e. icons, menu items and panels associated to resources to explore/manage and according to end-user roles. Reactions associated to these described graphical elements can be implemented by Java classes or BeanShell scripts which must be conform to the plug-in programming interface. Finally, the explorer framework implements the interpretation of explorer configurations and executes plug-in classes/scripts according to users' interactions. This framework is implemented as an extensible set of software components conform to the ObjectWeb Fractal component model defined by INRIA and France Telecom. Moreover a set of plug-ins is already provided to explore and manage any Java objects and Fractal components.

Fractal Explorer is already reused and customized by our Apollon, FAC, GoTM, and OpenCCM software to provide respectively explorer consoles for XML documents, Fractal aspect components, component-based transaction services and CORBA objects/components.

Fractal Explorer is a LGPL open source software available at http://fractal.objectweb.org.

## 5.3. GoTM

**Keywords:** *Component-Based Software Framework, Middleware Transaction Services*.

**Participants:** Philippe Merle, Romain Rouvoy [correspondant].

GoTM is a Fractal component-based software framework to build middleware transaction services.

GoTM is composed of an extensible set of Fractal components providing basic building blocks (Transaction, Resource, Coordination, Concurrency, etc.) to build various transaction models and services (OMG OTS, Sun JTA, etc.) [103]. This framework provides personalities to build Java Transaction Services (JTS) and Object Transaction Services (OTS). To deal with the transaction demarcation concern, GoTM provides the Open Transaction Demarcation Framework (OTDF) as a dedicated personality. The JTS and OTDF personalities are currently integrated into the ObjectWeb JOnAS application server.

The GoTM framework can be used to build heterogeneous transaction services [103]. Heterogeneous services support multiple transaction standards simultaneously without the performance degradation implied by the use of a coordination protocol.

The GoTM framework implements various 2-Phase Commit protocols (e.g. 2PC, 2PCPA, 2PCPC, etc.). It dynamically adapts the active transaction protocol to provide better transaction completion time depending on the commit/abort rate statistics [40].

The GoTM component-based software framework is designed on top of the ObjectWeb Fractal component model and is implemented on top of the ObjectWeb Julia reference implementation. Since recently, GoTM uses the AOKell software supported by the Jacquard project. AOKell implements the Fractal component model using Aspects Oriented Programming (AOP). This choice provides to GoTM the complementarity of

technologies like aspects and components to build middleware solutions with added values. GoTM uses also the Fractal Explorer software to build its management tool. GoTM provides various explorer plugins (GoTM, JTS, OTS, etc.). Thus, GoTM consoles can manage heterogeneous GoTM transaction services. Finally, GoTM OTS personality aims to be integrated in the OpenCCM software developed in the Jacquard project.

On the topic of Transaction and Components, GoTM and AOKell projects are also collaborating to provide transparent transaction support to Fractal components.

GoTM results from the activities done in the Transaction Working Group created in the context of the ITEA OSMOSE project. GoTM will be integrated in the next Enterprise Service Bus (ESB) solution developed by the ObjectWeb consortium in the context of the RNTL JOnES project.

GoTM is a LGPL open source software available at http://gotm.objectweb.org.

## 5.4. OpenCCM

**Keywords:** *CORBA Component Model*, *Component-Based Middleware*.

**Participants:** Areski Flissi, Philippe Merle [correspondant].

OpenCCM is a middleware platform for distributed applications based on CORBA components.

OpenCCM stands for the Open CORBA Component Model Platform: The first public available and open source implementation of the CORBA Component Model (CCM) specification defined by the Object Management Group (OMG). The CORBA Component Model (CCM) is the first vendor neutral open standard for Distributed Component Computing supporting various programming languages, operating systems, networks, CORBA products and vendors seamlessly. The CCM is an OMG's specification for creating distributed, server-side scalable, component-based, language-neutral, transactional, multi-users and secure applications. Moreover, one CCM application could be deployed and run on several distributed nodes simultaneously.

OpenCCM allows users to design, implement, compile, package, assemble, deploy, install, instantiate, configure, execute, and manage distributed CORBA component-based applications. For these purposes, OpenCCM is composed of a set of tools, i.e. UML and OMG IDL model repositories, compilers, code generators, a graphical packaging and assembling tool, a distributed deployment infrastructure, extensible containers integrating various services (communication, monitoring, transaction, persistency, security, etc.), and a graphical management console.

OpenCCM is a LGPL open source software available at http://openccm.objectweb.org.

## 5.5. Java Aspect Components

**Keywords:** *Java Aspect Components*, *dynamic weaving*.

**Participant:** Renaud Pawlak [correspondant].

JAC (Java Aspect Components) is a project consisting of developing an aspect-oriented middleware layer. JAC current version is 0.12.1. Current application servers do not always provide satisfying means to separate technical concerns from the application code. Since JAC uses aspect-orientation, the complex components are replaced by POJOs (Plain Old Java Objects) and technical concerns implementations that are usually wired deep into the containers implementations are replaced by loosely-coupled, dynamically pluggable aspect components. JAC aspect components provide: seamless persistence (CMP) that fully handles collections and references, flexible clustering features (customisable broadcast, load-balancing, data-consistency, caching), instantaneously defined users, profiles management, access rights checking, and authentication features. See http://jac.objectweb.org

## 5.6. SafArchie Studio

**Keywords:** *Transformation process*, *architecture-centric IDE*.

**Participants:** Olivier Barais [correspondant], Missi Tran-Anh.

SafArchie Studio is a software architecture-centric IDE that implements the SafArchie (Safe Architecture) proposition. It is composed of three main parts. The first one is a set of extensions for ArgoUML to transform the UML case tool into a software architecture case tool. It allows the architect to graphically design a component based software architecture. The second part analyses the software architecture consistency. This consists of checking the structural compatibility between bound ports and the behavioral compatibility between bound components with respect to the SafArchie model. The third part is a skeleton generator for the ArchJava language or the Fractal component model (more precisely the France Telecom's official implementation: Julia). This generator provides a first level connection between the design and the implementation of component based system. The longer term goal of SafArchie Studio is to provide a unified IDE to design, transform and control the evolution of component based applications. The work on SafArchie Studio is a part of Olivier Barais's PhD thesis and is mainly developed by Missi Tran-Anh and Olivier Barais.

TranSAT tools are designed as extensions of SafArchie Studio (see above). They aim at providing IDE to allow the architect to design a software architecture stepwise. TranSAT is associated with three main tools

- new diagram editors to specify new software architecture patterns
- a compiler for the software architecture pattern. It performs static analyses to guaranty that the pattern will not compromise the consistency of the software architectures that will be modified.
- a transformation processor. It integrates a new pattern into a software architecture.

The transformation processor includes a first module to find all the join points that satisfy the join point mask constraints, a second module to dynamically verify the consistency of the resulting software architecture and finally a transformation engine to perform the transformations on the join points selected by the architect. We have three implementations for the processor, each of which uses a specific tool: AGG based on the graph theory [108], [47], CIAO Prolog based on the predicate logic [58], and DROOLS based on the production rules [56].

The longer term goal of TranSAT tools consists of comparing the different technologies used to implement an efficient transformation operator. Furthermore, we are going to experiment applying the TranSAT's transformation process to other kinds of diagram such as class diagram or deployment diagram. The work on TranSAT is a part of Olivier Barais's PhD thesis. It is mainly developed by Missi Tran-Anh and Olivier Barais. See at http://transat.gforge.inria.fr

## 5.7. FAC

**Keywords:** *AOP*, *Fractal*.

**Participants:** Nicolas Pessemier, Lionel Seinturier [correspondant].

FAC (Fractal Aspect Component) is an extension of the Fractal component model for aspect-oriented programming. The purpose of FAC is to provide a programming model where components and aspects are first-class entities which can be assembled and composed seamlessly by designers and developpers. FAC is innovating in the sense that, on the international stage, aspects have been studied so far only at the object level. We believe that the issues of code tangling and scattering exist at a higher level of granularity, that of components. With FAC, we wish to provide one of the first model and its implementation, for modularizing crosscutting concerns in component-based applications. The longer term goal of FAC is also to provide an unified framework where crosscutting concerns can be modularized at different levels of granularity: object, component and architecture. The work on FAC is conducted in the context of a France Telecom R&D research grant and is the subject of Nicolas Pessemier PhD thesis. See at http://www.lifl.fr/ pessemie/FAC/

## 5.8. AOKell

**Keywords:** *AOP*, *Fractal*.

**Participants:** Nicolas Pessemier, Lionel Seinturier [correspondant].

AOKell is a reflective and open implementation of the Fractal component model. So far, software components are used in various application domains (for embedded systems to e-commerce web sites). However the model of a component is different in each of these domains with solutions such as EJB, CCM, .Net, Accord/UML, ArchJava, OpenCOM, K-Component, etc. This profusion hinders the broad adoption of component based techniques. The challenge we are addressing with AOKell is to design and implement a component model adaptable to various application domains. To achieve this goal, AOKell is a reflective component model with two dimensions: the business dimension and the control dimension. The former is concerned with the programming of the application functionalities, while the latter is concerned with the control and the technical services needed by the application. The business dimension is fully programmable with AOKell with the same artefacts (component, binding, components assembly) which are available for the business dimension. The integration of these two dimensions is achieved with aspect-oriented programming. The work on AOKell is conducted in the context of a France Telecom R&D research grant.

http://aofractal.gforge.inria.fr/api/aokell/1.0/aokell/overview-summary.html

## 5.9. Spoon

**Keywords:** *Program transformation*.

**Participants:** Renaud Pawlak [correspondant], Nicolas Petitprez.

Spoon is an annotation-driven program transformation tool, which is built on SUN's APT (Annotation-Processing Tool) and provides pure Java templates for easy and type-safe transformations. Spoon integrates a framework (based on the visitor pattern) that allows the processing of any kind of Java element, which are reified in a meta-model. Thanks to generics and annotations, it is straightforward for the programmer to define any kind of program processor that is type-safe and that processes a particular program or a particular annotation. A processor can validate a program or transform it. For the program transformations, we are able to introduce a pure Java and type-safe template mechanism to define the transformations (this is possible only with Java 5 generics). This template mechanism is less powerful than the C++ one, but it is fully typed and totally supported by the IDE. As such, it is far easier to use for the programmers. The Spoon sofware is available on the INRIA GForge at http://spoon.gforge.inria.fr.

## 5.10. UML Profile

**Keywords:** *CCM Specification , UML Profile*.

**Participant:** Olivier Caron [correspondant].

This software enables to design a UML model corresponding to the standardized UML profile for CCM. It both checks the validity of the model and generates IDL description. This software is available as a UML Objecteering module at the following address: http://www.lifl.fr/jacquard/software/Profil-UML-CCM/index.html.

## 5.11. OCL Support

**Keywords:** *Eclipse Modeling Framework, OCL support* .

**Participant:** Gilles Vanwormhoudt [correspondant].

This project extends the Eclipse Modeling Framework (EMF) with an OCL support. EMF is a set of development tools that provides a metamodeling approach for the Eclipse environment based on Essential MOF. It enables the definition of metamodels and generates Java implementation of theses metamodels, which both support the creation and manipulation of model instances. The proposed extension gives the abilities to attach OCL constraints (pre/postcondition, invariant) to any metamodel defined with EMF, to verify their coherence and to translate these contraints into Java code for evaluating them on instance models. Thanks to the OCL capabilities, developpers exploiting EMF can specify their models and metamodels in

a more precise and coherent way. This software is available as an Eclipse plugin at the following address: http://www.enic.fr/people/Vanwormhoudt/siteEMFOCL/index.html.

## 5.12. COCOA Modeler

**Keywords:** *Parameterized Model Application, Targeting Strategies, UML Case Tool.*

**Participant:** Alexis Muller [correspondant].

Cocoa Modeler is a modeling tool based on the Eclipse Modeling Framework and the UML2 Eclipse plugin. It adds a graphical representation and allows defining generic models as UML2 template packages. It provides the functionality of composing these generic models and applying them in order to build a complete system. This work in progress includes different code generation strategies. Cocoa Modeler is available at the following address: http://www.lifl.fr/ mullera/cocoamodeler.

# 6. New Results

## 6.1. Open Middleware for the CCM

### *6.1.1. Component Middleware for Ubiquitous Computing*

**Participants:** Areski Flissi, Philippe Merle.

Multiplication of mobile devices (laptop, PDA, smartphone, etc.) and generalized use of wireless networks imply changes on the design and execution of distributed software applications targeting ubiquitous computing. Many strong requirements have to be addressed: heterogeneity and limited resources of wireless networks and mobile devices, networked communications between distributed applications, dynamic discovery of services and automatic deployment on mobile devices.

In [26], [25], [24] and [5], we present our approach to address these challenges, and the design and implementation of the OpenCCM Mosaiques framework.

The OpenCCM Mosaiques' framework is a component-based software infrastructure to design, discover, deploy, and execute ubiquitous contextual services, i.e. distributed applications providing services to mobile end-users but only available from a particular place. These ubiquitous contextual services are designed as assemblies of distributed software components.

The OpenCCM Mosaiques' infrastructure allows mobile end-users to dynamically discover the mobile assemblies of ubiquitous contextual services according to the end-users physical location and also hardware/software device capabilities. It is based on a multicast discovery protocol that reduces power consumption and network traffic and a negotiation protocol to present to end-users only the mobile assemblies that are adapted to their device capabilities. Next, the OpenCCM Mosaiques' infrastructure allows end-users to automatically deploy the mobile assemblies of ubiquitous contextual services on their own devices, and manages the lifecycle of the services (i.e. cache management of mobile component assemblies for future re-use, uninstallation of services, etc.).

The OpenCCM Mosaiques infrastructure is implemented using the OMG CORBA Component Model (CCM), on top of the OpenCCM open-source platform.

### *6.1.2. Model-driven Approach to Build Component Middleware Deployment Infrastructure*

**Participants:** Jérémy Dubus, Areski Flissi, Philippe Merle.

Most of the component middleware allows now the automatization of applications deployment process. The software that is responsible of the execution of the deployment process (called deployment machine) instantiates the application from its architectural description. This latter describes a concrete configuration of a component-based application. Each applicative configuration contains a set of concepts that are related to the different component models, and can be expressed using various architectural description languages (ADL).

For instance, as far as the OMG CORBA Component Model (CCM) is concerned, the ADL used is a XML-based one (which files are called CCM descriptors), and some of the concepts are: home, component, instance, binding, business component property, placement, etc. As another example, the ObjectWeb Fractal component model uses similar concepts (e.g. factory, component, instance, etc.) plus some concepts that are not present in the CCM specification, such as composite / primitive component for example, and its own ADL to describe configurations. Upstream of the deployment process, the concrete applicative configuration is parsed, using a front-end adaptor for the used ADL. Then, thanks to a transformation, an abstract deployment model reifying the concrete configuration is obtained. The deployment machine engine next maps the reified applicative configuration on the targeted platform and instantiates the application, using back-end adaptors wrapping the platform deployment API.

Nevertheless, these deployment APIs are specific to middleware platforms. Thus, deployment machine implementations are bound to middleware platforms. There is no capitalization of deployment concepts, software code or methodology for building these deployment infrastructures. The idea then is to apply the OMG MDA (Model Driven Architecture) approach and make abstraction efforts on deployment process for being independent of both 1- concepts and languages used to describe applicative configurations and, 2- middleware execution platforms (and the associated deployment API).

This approach, presented in [27], introduces a workflow metamodel which allows us to define deployment models independently of any targeted component middleware (i.e. a PIM - Platform Independent Model). Indeed, deployment of component-based applications consists of executing an ordered list of basic deployment tasks such as uploading component binaries to the execution sites, loading them in memory, instantiating components from their factories, interconnecting their ports, configuring business and technical properties and final activation of components. These tasks have to be scheduled, coordinated and executed according to a defined order. Each workflow process activity corresponds then to an elementary deployment task. Such a model is then refined (with component model concepts) for each targeted middleware and the obtained model is mapped (following some transformation rules between PIM and PSM metamodels concepts) to a Platform Specific Model (PSM). Finally, the last step is the mapping (according to some generation rules) to a technological execution platform. The software that implements the deployment machine is generated for a specific execution platform. The models and transformations of this approach are illustrated on a CORBA Components deployment machine implemented using the Fractal component model.

In order to validate this approach, in [45] we have studied related works in the deployment domain. We have shown the every component platform has its own deployment machine, especially because it has its own deployment model. The second point that is to be pointed out, is that current research focusing on deployment domain is spread across the deployment life cycle. There is no integration of any deployment concerns (such as placement of component, architectural rules, dynamic physical architecture discovery, etc.) in current deployment environments/machines. So the proposition presented in [45] emerges from those two points.

The proposal is a deployment environment based on the Deployment and Configuration specification published by the Object Management Group (OMG). This specification provides a deployment meta-model independant from any component platforms, this meta-model is then used in our deployment environment to represent the deployment model. Many refinements can then be applied to this model, in order to weave any deployment concerns. This process is strongly inspired by the Model Driven Engineering paradigm. The final refinement of this model consists of a transformation from that deployment meta-model, to the deployment task meta-model provided in [27]. Executing those generated tasks starts the effective deployment process.

A first personnality of this environment has been developped which allows to deploy OpenCCM Components from deployment plans expressed as OMG D&C descriptors. In addition of being more elegant, this deployment process has shown better performance than the current deployment machine shipped with OpenCCM, i.e. OpenCCM Distributed Component Infrastructure (DCI).

### *6.1.3. Component-Based Software Framework for Building Transaction Services*

**Keywords:** *Component-Based Software Frameware, GoTM, Middleware Transaction Services.*

**Participants:** Philippe Merle, Romain Rouvoy.

GoTM is a component-based software framework for building middleware transaction services.

*6.1.3.1. Overview of the GoTM Activity*

Transactions have always been involved in various applications since they have been introduced in databases. Many transaction services have been developed to address various transaction standards and various transaction models. Furthermore, these transaction services are more and more difficult to build since the complexity of the transaction standards is increasing constantly. Each transaction service implements pieces of code that have already been written in another transaction services. As a consequence, there is no code factorization between the transaction services and the added values of each transaction service, such as extensibility or performance, are never reused in another transaction service.

In [103], we present GoTM, a Component-Based Adaptive Middleware (CBAM) software framework. It can be used to build various transaction services that are compliant with existing transaction standards (OMG OTS, Sun JTS, etc.). GoTM provides adaptive properties to support different transaction models and standards in the same transaction service. GoTM supports also the definition of new transaction models and standards as new components of the framework. Finally, GoTM provides (re)configurability, extensibility and adaptability properties as added values. The implementation of the GoTM framework is based on the Fractal component model. The next sections illustrate two experiences performed this year with the GoTM framework.

*6.1.3.2. Building Heterogeneous Transaction Services*

The diversity of transaction services leads to compatibility problems among applications using different transaction standards. This compatibility usually ensures that transaction services can cooperate in a system. To deal with this issue, current trends use coordination protocols. Coordination protocols are responsible for synchronizing the execution of transaction services based on different transaction standards. Nevertheless, these protocols can be intrusive and often introduce an additional complexity to the system.

In [103], we present an approach to build an *Adapted Transaction Service*, named ATS, which supports several transaction standards concurrently. The objective of ATS is to facilitate the transaction standards composition. To introduce ATS we detail how the *Object Transaction Service* (OTS), *Web Services Atomic Transaction* (WS-AT) and *Java Transaction Service* (JTS) standards can be composed. For this, the OTS, WS-AT and JTS interfaces are analyzed and the required/provided functions are identified. The functions are specialized in strategies to implement transaction standard semantics. ATS is built by composition of these strategies and adapters. Adapters ensure the compliance with transaction standards interfaces. Besides, the ATS implementation is introduced, which uses the GoTM framework and the Fractal component model. GoTM is a software framework that provides various transactional components.

We show that this approach does not introduce an additional overhead to legacy applications and supports well scalability. Moreover, this approach can be easily extended to support additional transaction standards. Future work will investigated the definition of personalities for Web Services Transaction and Activity Services.

*6.1.3.3. Supporting dynamic adaptation of 2-Phase Commit protocols*

For years, transactional protocols have been defined to address specific application needs. Traditionally, when implementing a transaction service, a protocol is chosen and it remains the same during the system execution. Nevertheless, the dynamic nature of nowadays application contexts (e.g., mobile, ad-hoc, peer-to-peer) and behavioral variations (semantic-related aspects) motivate the needs for application adaptation. Next generation of system applications should be adaptive or even better self-adaptive. In [40], we propose (1) a component-based architecture of standard 2PC-based protocols and (2) a self-Adaptive Component-based cOmmit Management, named ACOM. Self-adaptation is obtained by behaviour awareness and component-based reconfiguration. This allows ACOM to select the most appropriate protocol according to the context. We show that using ACOM performs better than using only one commit protocol in a variable system and that the reconfiguration cost can be negligible.

Future work will investigate the design of a dedicated high-level model to describe transaction validation protocols.

### *6.1.4. Middleware Benchmarking*

**Keywords:** *Benchmarking, CCM, CORBA, EJB, Java-Based Middleware, ORB, Round-Trip Latency, Web Services*.

**Participants:** Christophe Demarey, Cédric Dumoulin, Philippe Merle.

This work targets to benchmark various heterogeneous middleware platforms in order to help applications' designers to select the right platform according to their applications' performance requirements.

Nowadays, distributed Java-based applications could be built on top of a plethora of middleware technologies such as Object Request Brokers (ORB) like CORBA and Java RMI, Web Services, and component-oriented platforms like Enterprise Java Beans (EJB) or CORBA Component Model (CCM). Choosing the right middleware technology fitting with application requirements is a complex activity driven by various criteria such as economic costs (e.g. commercial or open source availability, engineer training and skills), conformance to standards, advanced proprietary features, performance, scalability, etc. Regarding performance, a lot of basic metrics could be evaluated as round-trip latency, jitter, or throughput of twoway interactions according to various parameter types and sizes.

Many projects have already evaluated these middleware performance metrics. Unfortunately, they have not compared different kinds of middleware platforms simultaneously. This could be helpful for application designers requiring to select both the kind of middleware technology to apply and the best implementation to use.

In [4], we present an experiment report on the design and implementation of a simple benchmark to evaluate the round-trip latency of various Java-based middleware platforms, i.e. only measuring the response time of twoway interactions without parameters. Even if simple, this benchmark is relevant as it allows users to evaluate the minimal response mean time and the maximal number of interactions per second provided by a middleware platform. Empirical results and analysis are discussed on a large set of widely available implementations including various ORB (Java RMI, Java IDL, ORBacus, JacORB, OpenORB, and Ice), Web Services projects (Apache XML-RPC and Axis), and component-oriented platforms (JBoss, JOnAS, OpenCCM, Fractal, ProActive). This evaluation shows that our OpenCCM platform already provides better performance results than most of the other evaluated middleware platforms.

## 6.2. Aspect-oriented Design of Dynamic Components Assemblies

### *6.2.1. Formal models and approaches*

**Participants:** Olivier Barais, Dolorès Diaz, Laurence Duchien, Renaud Pawlak, Lionel Seinturier.

With new component platforms, architects create distributed applications by assembling components. For all these platforms, a software architecture defines the application organization as a collection of components plus a set of constraints on the interactions between components. To face the difficulties for building correct software architecture, abstract software architecture models were defined. They are powerful methods in the specification and analysis of high-level designs. Lots of architecture description models have been defined to describe, design, check, and implement software architectures. Many of these models support sophisticated analysis and reasoning or support architecture-centric development.

Nevertheless, these models are often static or work only on the components composition. In these conditions, it is difficult to build large software architecture or to integrate new components in an existing software architecture or to add a forgotten concern such as security or persistency. Thus, we propose SafArchie, an abstract component model for designing software architectures [54]. With SafArchie, we based our approach on architecture types that are points of reference at each step of our reasoning [54], [52]. We develop SafArchie Studio [53] [1], [11], an architecture centric tool based on three-view perspective and driven by the component life cycle.

Still related to the domain of static validation of component-based architectures, we have also worked in validating architectures composed by the means of aspects. CompAr (Composing Around advice) is a language that has been created to automatically detect and solve Aspect-Composition Issues (ACIs) of around advice.

When doing AOP or any related programming technique using around advising, such as interception-based approaches, an ACI can arise when several around pieces of advice apply to the same advised element (it is then an advice chain). Indeed, each around advice code holds some implicit execution contraints; when they are composed together in a chain, they can conflict in such a way that the global system does not fulfill some of the implicit contraints anymore. With CompAr, one can specify the execution constraints of the advice as well as an abstract representation of its code. For a given advice chain order, the CompAr compiler checks that all the execution contraints are fullfilled by evaluating the abstract specification for all the possible contexts. In the case an execution constraint is not fulfilled, the compiler reports an error and some indications to the user so that s/he can solve the ACI, either by reordering the chain, or by changing the abstract specification of some advice. CompAr and some examples of conflict resolving have been published in [37].

### *6.2.2. Aspects and Components for Software Architectures*

**Participants:** Laurence Duchien, Frédéric Loiret, Nicolas Pessemier, Lionel Seinturier.

This action is concerned with the definition and the implementation of a programming environment for complex applications in the domains of middleware, embedded systems and ubiquitous computing. This environment investigates the use of two software engineering techniques: component-based software engineering (CBSE) and aspect-oriented software development (AOSD).

In a post-object world (object in the sense of object-oriented programming languages), both techniques are recognized as being solutions to manage the complexity of software. Far from being redundant or conflicting, they are complementary in the sense that CBSE is a solution for defining the functional part of a software architecture, whereas AOSD allows modularizing the crosscutting, non functional, part of this software.

Two new results have been achieved in 2005 around this action:

- The definition of a general framework [39] for integrating AOSD and CBSE. We showed that both concepts can be integrated at various level of granularity: the level of the software architecture, the level of the individual building blocks of this architecture, and the level of the implementation of these building blocks. These three levels have different purposes, and are jointly needed when considering applications with components and aspects. This result is part of the ongoing work of Nicolas Pessemier's PhD thesis.
- Component frameworks (e.g. EJB, CCM, .Net/COM+) provide an infrastructure for hosting component-based applications. Various services are provided to manage components. The granularity and the features included in these services vary because each component model addresses needs that may be different. However, so far, component models are mostly closed, black boxes, in which services are hardcoded and can not be changed. We showed that the engineering of these services could be opened with components. We have thus proposed a two-dimension component model where the business dimension is concerned, just like in the other component models, with the functional part of the application, whereas the control dimension is concerned with the services provided by the framework to host and manage the components of the business dimension. Through this approach, we propose a high-level solution for engineering the control part of software components. Several domains may benefit from this approach, such as the domains of self-healing, self-testable or proof-carrying software components. The proposed model and its implementation are called AOKell [13]. It conforms to the Fractal Specifications.

These two results are part of the ongoing grant with France Telecom R&D.

### *6.2.3. Transformation Languages and Tools*

**Participants:** Olivier Barais, Maja D'Hondt, Laurence Duchien, Julia Lawall, Anne-Françoise Le Meur, Carlos Francisco Noguera Garcia, Renaud Pawlak, Nicolas PetitPrez, Missi Tran-Anh.

Building a software architecture with all the needed concerns at once is a complex task. Consequently, an incremental approach where each concern is successively integrated into the architecture will be preferred. Unfortunately, the existing architecture description languages, which enable an architect to create an architecture by constructing and combining increasingly complex elements, are not suitable to cleanly structure concerns that crosscut the software architecture, such as security. To address this issue we have proposed TranSAT, which is a framework to specify incremental software architecture transformations [2], [21], [11]. TranSAT isolates the description of each concern in a separate architecture construct, the *pattern*, that is automatically integrated with an existing software architecture by a *weaver*. Analogous to an aspect, a pattern consists of the new architecture fragment to be integrated, a description of where it can be applied, and a specification of the transformations that should be performed to connect it to the existing architecture. As a pattern can specify updates to multiple elements in an existing architecture, it is suitable for expressing crosscutting concerns. We have designed the language for specifying patterns in order to enable verifications that ensure that the concern is safely integrated into the existing architecture. The verification comprises static verifications that check coherence properties once the pattern has been defined, and dynamic verifications that focus on the parts of the architecture that are affected by the pattern. This work has been done in the context of the PhD of Olivier Barais [21], [11].

In order to support analysis and design-level validation in an efficient way, we are also currently working at building language-level transformation tools. For this, we have developped Spoon, which is an annotation-driven program transformation tool. In 2004, SUN released Java 5, which includes "generics" and "annotations". With annotations, the Java 5 programmers can define and attach metadata on the program elements. These metadata allow for declarative configuration of the programs to define non-functional properties. In many cases, annotation-based configuration is preferred over typical XML-based one, because it is more integrated into Java, much safer (type-safety), and it avoids information redundancy. However, current tools to deal with annotations are currently limited to large-grained annotation-based program validation and documentation generation. Spoon's goal is to answer the needs of the industry and of the Java community in general for a tool that fully supports generics and annotations and that allows for fined-grained and type-safe program validation and transformations. Spoon has applications in several domains. Spoon can be used in component containers (to implement annotation-driven deployment), for general software engineering (in particular AOSD can benefit from the use of annotation to parameterize the pointcuts), ubiquitous computing (annotations would help the programs to target the right environment), real-time systems (programmers would then define time constraints annotations directly in the Java code), etc. Spoon and its application to AOSD has been published in [38].

## 6.3. Functional Aspects

**Participants:** Olivier Caron, Bernard Carré, Raphaël Marvie, Alexis Muller, Gilles Vanwormhoudt.

Building new systems by composing pre-built and validated models promises a quicker design of more reliable systems. Several model driven approaches recognize templates, particularly in the UML sphere, as a powerful technique to specify parameterized models and their usage in the construction of whole system models [67], [64], [102], [106].

We focus on how to define model driven engineering construction processes with such parameterized models. For this, we need to express complex compositions of parameterized models, which must be applied in a coherent way. Such building processes raise open issues: Is the result influenced by the application order ? Can we compose independent parameterized models ? Is it possible to define composition chains and find equivalent ones that express the same resulting model ?

To support such processes, we introduce an operator (*apply*) to express the application of a template to an existing model [7]. This operator specifies how to obtain a model from an existing one by the application and

composition of generic ones. It also allows the combination of generic models in order to design richer ones. Thanks to this operator, sequences of applications can be expressed in a coherent way. Alternative composition sequences of parameterized models can be elaborated to build the same system model.

We have formalized the semantics of this operator and proved some properties that guarantee the correctness of application chains and their alternative ordering capacities. Our formalization is deliberately independent from any specific usage or existing methodologies. This gives the ability to provide strategies to transform composition of parametrized models into platform specifc models.

At a practical stage, we are developping a design tool based on the Eclipse Modeling Framework extended with the OCL support [43] and the Eclipse UML2 plugin. Our tool called Cocoa Modeler allows defining generic models as UML2 template packages and provides the functionality of composing these generic models and applying them in order to build a complete system. Strategies to transform composition of parameterized models into platform specific models are also offered. This tool will give the ability to manage libraries (design, composition, import,...) of parameterized models as standard UML2 templates.

Decomposing a system architecture into views has also been studied in the definition of system architectures [84]: after identifying the core concepts of a domain, concerns are specified one by one before explaining their integration. In the continuation of this work, we have studied the definition of modeling processes [44]. In this second stage we have proposed a framework for assisting the user in his/her modeling activity[32]. First, the modeling process is defined as steps with a subset of the modeling operations available and constraints that states when the step can be started and considered terminated. Second, based upon this specification the modeling environment controls the user activity: providing concern modeling only when possible, and evaluating the modeling constraint to drive the move between modeling steps. We have studied this approach both in the context of iterative development and software evolution of 3-tiers applications. Finally, this work is a source of collaboration with the Triskell Research Project (IRISA) as we are implementing our proprosal using KerMeta [109]. This implementation is intended to contribute to the KerMeta environnement in the *Modelling Processes* sub-project.

# 7. Contracts and Grants with Industry

## 7.1. France Telecom

**Participants:** Laurence Duchien, Nicolas Pessemier, Lionel Seinturier.

This contract is a CRE ("Contrat de Recherche Externe") that takes place in the context of the "accord-cadre" between INRIA and France Telecom R&D. This is a 3-year contract that begun in October 2004. The scientific teams involved in the project are for INRIA, the Jacquard project-team, and for France Telecom R&D, the ASR/Polair department. The contract goal is to study and construct component and aspect based software architectures. The Fractal component model from France Telecom R&D and the JAC AOP framework from Jacquard form the background of this work. The expected result is a model (FAC for Fractal Aspect Component) that merges and unifies aspects and components. Nicolas Pessemier PhD thesis work is directly related to this contract.

## 7.2. NorSys

**Participants:** Dolorès Diaz, Laurence Duchien, Lionel Seinturier.

This contract is associated to a CIFRE PhD thesis between the Jacquard project-team and the NorSys service company. The goal of the contract is to study the aspect-orientation in the early stages of software development. AOP emerged as a programming technique but the question is now open in the international research company to tell whether it can also bring to innovations into the early stages of requirement engineering, analyze and design. This contract begun in January 2004. Dolores Diaz PhD thesis work is directly related to this contract.

# 8. Other Grants and Activities

## 8.1. Regional Initiatives

### *8.1.1. IRCICA*

**Participant:** All the project.

The 'Region Nord Pas de Calais' has initiated a large research plan around the new technologies for communications. We lead the software section of this plan. Beyond this plan the 'Region Nord Pas de Calais' has facilitated the creation of a new research institute called IRCICA to promote new collaborative research projects between software and hardware laboratories. The Jacquard project is one of the first projects supported by this institute.

### *8.1.2. MOSAIQUES*

**Participant:** All the project.

The MOSAIQUES Project ("MOdèles et InfraStructures pour Applications ubIQUitairES" or Models and middleware for ubiquitous applications) defines a design and programming framework for application definitions that run in ubiquitous environment. The project includes The University of Lille with LIFL Laboratory (STC and SMAC teams) and INRIA projects Jacquard and POPS, TRIGONE laboratory, INRETS, Ecole des Mines de Douai and The University of Valenciennes and of Hainaut-Cambrésis. Application domains are transportation and e-learning systems. Laurence Duchien is in charge of this project.

### *8.1.3. FLEXIBLE*

**Participants:** Renaud Pawlak, Anne-Françoise Le Meur.

The goal of the FLEXIBLE BQR project, which is supported by the University of Lille 1, is to develop a platform to build adaptable and robust component-based applications. Applications built within this framework will be able to be safely adapted with respect to their changing environment. Adaptation mechanisms will mainly rely on aspect-oriented programming technics. The robustness of the adaptation process will be obtained through the use of DSLs to describe the contracts and properties that the application must respect as it evolves.

## 8.2. National Initiatives

### *8.2.1. AS MDA*

**Participants:** Laurence Duchien, Raphaël Marvie.

The specific action MDA, created in June 2003 and funded by CNRS, studies the interest of the Model Driven Architecture approach. The standard promoted by OMG is only one example. The aim of this AS is to organize the research community in this domain in order to understand and to help the industrial community in an approach that it can be a significant evolution on the middle and long term. This AS includes IRIN, LIFL, LSR, IMAG, I3S, PRIM and CEA laboratories. We study the platform definition in this MDE context [19].

### *8.2.2. ARC COA*

**Participants:** Philippe Merle, Lionel Seinturier, Renaud Pawlak, Nicolas Pessemier.

This 2-year project is funded by the INRIA Cooperative Research Initiative (ARC) whose partners are the PARIS, SCALAPLIX and Jacquard Project-Teams. Its objective is to design an experimental platform allowing dynamic adaptation and steering of distributed numerical simulation applications using aspect weaving technics on top of component models.

## 8.3. European Initiatives

### *8.3.1. ObjectWeb*

ObjectWeb is an European initiative to promote high quality open source middleware. The vision of ObjectWeb is that of a set of components which can be assembled to offer high quality. We are member of this consortium, and Fractal Explorer, GoTM, JAC, and OpenCCM are projects hosted by the consortium.

### 8.3.2. ITEA OSMOSE

**Participants:** Christophe Contreras, Philippe Merle, Jérôme Moroy, Romain Rouvoy.

OSMOSE stands for 'Open Source Middleware for Open Systems in Europe'. It is an ITEA project. The project groups 16 European industrials and 7 public labs. The goal is to give up an European dimension for the ObjectWeb consortium. The OSMOSE project wants to federate high quality components from European labs, and to produce applications for the great European industrial domains.

### 8.3.3. ITEA S4ALL

**Participants:** Christophe Contreras, Philippe Merle.

S4ALL ('Services-for-All') is an ITEA project whose partners are Alcatel CIT, Bull, Capricode, Fraunhofer Fokus, HIIT, INRIA, Instituto de Telecomunicaçoes, INT, mCENTRIC, Nokia, PT Inovaçao, Schneider Electric, Thales, Université Joseph Fourier, Universidad Politécnica de Madrid, University of Kassel, Vodafone, and Xquark/Odonata. The main vision is to build "A world of user-centric services that are easy to create, share and use". Our contribution in this project is to design and build a model driven infrastructure to automatically deploy J2EE components, OSGi services, and Web services for building distributed applications.

### 8.3.4. AOSD-Europe

**Participants:** Laurence Duchien, Lionel Seinturier, Renaud Pawlak, Olivier Barais, Alexis Muller, Carlos Francisco Noguera Garcia.

AOSD-Europe is an ongoing proposal to set up a Network of Excellence (NoE) on aspect-oriented software development within IST-FP6. The proposal brings together 11 research groups and among them members of the Jacquard project and other members from OBASCO, Pop-Art and Triskell INRIA projects. The proposal is led by Lancaster University, Darmstadt University and University of Twente. The goal of the NoE is to harmonise, integrate and strengthen European research activities on all issues related to aspect orientation: analysis, design, development, formalization, applications, empirical studies.

## 8.4. International Initiative

### 8.4.1. OMG

We work in the international consortium Object Management Group (OMG) since 1997. OMG defines well-known standards: CORBA, UML, MOF, MDA. We can quote our contributions to the OMG standardization work with the CorbaScript language (proposed to the Scripting Language for CORBA RFP, and accepted as the CORBA Scripting Language chapter of CORBA 3.x) and with the CORBA Component Model (CCM) chapter for which we lead the response group and the revision task force. We also participate in the definition of a UML profile for CORBA Components.

**Philippe Merle** is:

- Official representant of the INRIA at the OMG.
- Chair of the OMG Components 1.2 Revision Task Force (RTF).
- Member of the OMG Deployment Revision Task Force (RTF).
- Member of the OMG UML Profile for CCM Finalization Task Force (FTF).
- Member of submission team for the OMG UML Profile for CCM RFP.
- Member of the voting list for the OMG MOF 2.0 IDL RFP.
- Member of the voting list for the OMG MOF 2.0 Query/View/Transf. RFP.
- Member of the voting list for the OMG MOF 2.0 Versioning RFP.
- Member of the voting list for the OMG QoS for CORBA Components RFP.
- Member of the voting list for the OMG Streams for CCM RFP.

### *8.4.2. International cooperation with Rensselaer Polytechnic Institute*

**Participant:** Renaud Pawlak.

Funded by an INRIA "mini-action"- 9 weeks of researcher exchanges. Invited week of prof. Houman Younessi in Lille. The cooperation aims to built active synergies between INRIA and RPI around Use-Case driven analysis and design, and AOP-related techniques.

# 9. Dissemination

## 9.1. Scientific community animation

### *9.1.1. Examination Committees*

- **Jean-Marc Geib** was in the examination committee of the following PhD thesis :
  - Michael Hauspie, January 2005, Lille (adviser)
  - Jean-Charles Tournier, July 2005, Lyon (referee)
  - Haiwu He, July 2005, Lille,(chair)
  - Stéphane Bonnet, May 2005, Lille (adviser)
  - Mourad Alia, June 2005, Grenoble (examiner)
  - HDR Sylvain Lecomte, December 2005, Valenciennes,(referee)
  - HDR Nourredine Melab, November 2005, Lille (chair)
  - HDR Lionel Seinturier, December 2005, Paris (examiner)
  - HDR Antoine Beugnard, December 2005, Brest (examiner)
  - Arnaud Curruccu, November 2005, Lille (chair)
  - Patrick Tessier, December 2005, Lille (adviser)
  - Oussama Layaida, December 2005, Grenoble (referee)
- **Laurence Duchien** was in the examination committee of the following PhD thesis :
  - Mejdi Kaddour, March 2005, ENST (referee)
  - Colombe Herault, June 2005, University of Valenciennes (referee)
  - Marc Segura, July 2005, University of Nantes (referee)
  - Pierre-Charles David, July 2005, University of Nantes (chair)
  - Jérome Huguest, September 2005, ENST (referee)
  - Mikael Beauvois, September 2005, University of Grenoble (referee)
  - Cédric Teyssie, November 2005, University of Toulouse (referee)
  - Olivier Barais, November 2005, University of Lille (adviser)
  - German Vega, December 2005, University of Grenoble (referee)
  - HDR Jean-Marc Talbot, December 2005, University of Lille (chair)
  - Arnaud Bailly, December 2005, University of Lille (chair)
- **Philippe Merle** was in the examination committee of the following PhD thesis :

  - G. Haïk, May 2005, University of Paris 6 (chair)
  - M. Alia, June 2005, Institut National Polytechnique de Grenoble (referee)
  - N. Kouici, November 2005, University of Evry (referee)

- **Lione Seinturier** was in the examination committee of the following PhD thesis :

  - Fabrice Legond-Aubry, July 2005, CNAM (co-adviser)

- **Bernard Carré** was in the examination committee of the following PhD thesis :

  - M. Nassar, September 2005, Institut National Polytechnique de Toulouse

- **Raphaël Marvie** was in the examination committee of the following PhD thesis :

  - S. Bonnet, April 2005, University of Lille (co-adviser)

### *9.1.2. Journals, Conferences, Workshop*

- **Laurence Duchien** has been a member of the following program committees:

  - Program committee of the conference Langages, Modèles et Objets, Bern, March 2005,
  - Program committee of the conference MCETECH, Montréal, January 2005,
  - Program committee of the Journées Composants, Le Croisic, April 2005,
  - Program committee CFSE, Le Croisic, April 2005,
  - Program committee of the workshop on software evolution, Bern, March 2005,
  - Scientific committee of the summer school EIAH, Lille, July 2005,
  - Program committee of the conference IDM, Paris, July 2005.
  - Program committee JFPLA, Lille, September 2005.

- **Anne-Françoise Le Meur** has been a member of the following program committees:

  - Program Committee member of GPCE05, sep 29- oct 1 2005, Tallinn, Estonia

- **Philippe Merle** has been a member of the following program committees:

  - Program committee of DAIS 2005, 5th IFIP International Conference on Distributed Applications and Interoperable Systems, June 2005, Athens, Greece.
  - Program committee of JC 2005, 4th French Conference around Software Components, April 2005, Le Croisic, France.
  - Program committee of JDIR 2005, 7ième Journées Doctorales Informatique et Réseau, December 2005, Troyes, France.
  - Editorial board of the journal L'Objet.
  - Steering Committee of CNRS GDR ARS.

- **Renaud Pawlak** has been a member of the following program committees:

  - Reviewer for the journal "Transactions on Aspect-Oriented Software Development", LNCS, Springer-Verlag. Special Issue on Aspect-Oriented Programming for Systems Software and Middleware.
  - Program chair, workshop AOMD 2005 at the 6th ACM/IFIP/USENIX International Middleware Conference, November 2005, Grenoble
  - Editorial board, ADVICE Journal
  - Program committee member JFPLA, Lille, September 2005.

- **Lionel Seinturier** has been a member of the following program committees:

  - Program chair, JFDLPA 2005, 2ème Journée Francophone sur le Développement de Logiciels Par Aspects, http://www.lifl.fr/jfdlpa05/, September 2005, Lille
  - Editorial board, TSI, http://tsi.revuesonline.com/
  - Program commitee, 4ème Conférence Francophone autour des Composants Logiciels, April 2005, Le Croisic
  - Program commitee CDUR 2005, november 2005, Paris
  - Program commitee, workshop AOMD 2005 at the 6th ACM/IFIP/USENIX International Middleware Conference, November 2005, Grenoble

- **Miscellaneous**

  - **Romain Rouvoy** has been an external referee for the following program committees: DCCS 2005, International Conference on Dependable Systems and Networks.
  - **Raphaël Marvie** does consulting for companies like France Telecom (*Equipe Conseil CNRS*).
  - **Olivier Barais, Bernard Carré, Olivier Caron**, and **Raphaël Marvie** were external referees for the conferences LMO'2005 and IDM'2005 (1ères journées sur l'Ingénierie Dirigée par les Modèles).
  - **Philippe Merle** was external referee for Euro-Par 2005, August - September 2005, Lisboa, Portugal.
  - **Olivier Barais, Laurence Duchien, Lionel Seinturier, Nicolas Pessemier, Dolorès Diaz, Renaud Pawlak** were external referees for the conference GPCE 2005.
  - **Renaud Pawlak, Lionel Seinturier, Laurence Duchien** were external referees for the conferences ECOOP 2005.
  - **Maja D'Hondt** was organiser of the 2nd European Interactive Workshop on Aspects in Software (EIWAS2005), september 1st 2005, at Bruxelles (Vrije Universiteit Brussel).
  - **Wolfgang De Meuter** was co-chair of the Worshop sessions at ECOOP conference, july 2005, Glasgow.
  - **Lionel Seinturier** was the organiser of JFDLPA 2005 2ème Journée Francophone sur le Développement de Logiciels Par Aspects http://www.lifl.fr/jfdlpa05/ september 15 th 2005, Lille.
  - **Renaud Pawlak** was the organiser of the Workshop AOMD 2005 at the 6th ACM/IFIP/USENIX International Middleware Conference, November 2005, Grenoble

## 9.2. Teaching

**Jean-Marc Geib** teaches Object Oriented Design and Programming and Distributed Application Design in L3 and M1 at USTL, UFR IEEA.

**Laurence Duchien** teaches Distributed Applications Design - Master Professionnel Sciences et Technologies Mention Informatique - M1 and Master Professionnel Sciences et Technologies Mention Informatique - M2 - Spécialité IAGL et TIIR at USTL, UFR IEEA. She is in charge of the Master Professionnel Sciences et Technologies Mention Informatique - M2 - Spécialité IAGL at USTL, UFR IEEA.

**Anne-Françoise Le Meur** teaches Databases and the Internet, Design of distributed applications, and C programming at USTL, UFR IEEA.

**Raphael Marvie** is in charge of Object Oriented Design (Master 1 and 2) and Programming (Bachelor 3), Distributed Computing (Bachelor 3), Model Driven Engineering (Master 2 and Master Research), at USTL, UFR IEEA. He is co-responsible for Master 2 IAGL, assisting L. Duchien for projects and training periods.

**Bernard Carré** teaches OO Design and programming at Polytech'Lille Engineering School (USTL). He is in charge of the Computer Science and Statistics Departement of Polytech'Lille. He also teaches Software Technologies.

**Olivier Caron** is in charge of the following modules: Data Bases and Distributed Software Components at Polytech'Lille Engineering school. He also teaches Object-Oriented Programming and Operating Systems. He also manages the final year engineering students projects of the Computer Science and Statistics Departement of Polytech'Lille.

**Gilles Vanwormhoudt** teaches Algorithms and Programming in C, 1th year of the engineering program, Design and Programming of Distributed Applications, 3rd year of the engineering program and Technologies for Web development, 5th year of the engineering program, ENIC Telecom Lille 1. He is in charge of Multimedia computing and engineering specialization, 5th year of the engineering program, and the Project-Conferences-Report module for the "Multimedia computing and engineering" specialization, ENIC Telecom Lille 1.

**Maja D'Hondt** teaches Algorithms and Data structures in L1 computer science and L2 Mathematics at Bruxelles (Vrije Universiteit Brussel).

**Lionel Seinturier** teaches Middleware, components and aspects in Master pro IAGL Lille, Master recherche Lille, Master pro SAR Paris 6, Master recherche SAR Paris 6.

Jacquard team's members participate to some Research Masters of Computer Science (University of Lille, University of Paris 6, University of Montpellier, University of Valenciennes, ESSI in Nice and RPI University, Hartford, US) on the CCM, MDE and on AOP.

## 9.3. Administrative Responsibilities

**Jean-Marc Geib** is the head of the LIFL laboratory CNRS 8022, the head of the CIM axis, member of the UFR board at USTL, member of CSE 27nd section of Universities of Lille 1, Lille 2 and Littoral. He is chair of the management commitee TAC (Technologies Avancées pour la Communication) of the Etat-Region Contract Nord-Pas-de-Calais and the coordinator of the communication program since 2004. He is IRCICA cofounder (Institut de Recherche sur les Composants matériels et logiciels pour l'Information et la Communication Avancée) that is a research federation between LIFL (Science computing), l'IEMN (electronics) and the PhLAM (photonique). He is in charge of the RTP Distributed Systems of the CNRS STIC Dept. He is chair of the Réseau Thématique Prioritaire (RTP5) « Distributed Systems » of the CNRS STIC Department. He is head of the GOAL « Génie des Objets et Composants » team in LIFL Laboratory. He is member of the UFR board at USTL.

**Laurence Duchien** is member of the UFR board at USTL, member of the LIFL scientific board, member of CSE 27nd section of Universities of Lille 1, CNAM and Paris 6. She is member of scientific committee of the national ACI Security. She is also treasuser of ASF association (French chapter of ACM SigOps). She is member of ERCIM software evolution group.

**Philippe Merle** is in charge of the European Affairs within the Department for European and International Relations (DREI) for INRIA UR Futurs and is member of the Incitative Action Working Group (GTAI) of the Scientific and Technological Orientation Council (COST) of INRIA.

**Lionel Seinturier** is member of CSE 27nd section of university of Nanterre and of CNAM. He is member of Commission Nationale Universitaire (CNU) 27nd section. He is also member of the LIP6 scientific board.

**Romain Rouvoy** is member of the "Comité d'Unité de Recherche" (CUR) of UR Futurs and representing Ph. D. students.

**Areski Flissi** is member of the "Comité des Utilisateurs des Moyens Informatiques" (CUMI) of UR Futurs and representing Jacquard members.

**Bernard Carré** is member of CSE 27nd section of university of Valenciennes.

**Olivier Caron** is member of CSE 27nd section of university of Lille.

**Raphaël Marvie** is member of CSE 27nd section of university of Nice and is also member of the LIFL scientific board.

# 10. Bibliography

## Major publications by the team in recent years

[1] O. BARAIS, L. DUCHIEN. *SafArchie Studio: An ArgoUML extension to build Safe Architectures*, ISBN: 0-387-24589-8, Springer, 2005, p. 85–100.

[2] O. BARAIS, L. DUCHIEN, A.-F. L. MEUR. *A Framework to Specify Incremental Software Architecture Transformations*, in "31st EUROMICRO CONFERENCE on Software Engineering and Advanced Applications (SEAA 2005)", IEEE Computer Society, September 2005.

[3] O. CARON, B. CARRÉ, A. MULLER, G. VANWORMHOUDT. *A Framework for Supporting Views in Component Oriented Information Systems*, in "International Conference on Object-Oriented Information Systems, Geneva, Switzerland", Lecture Notes in Computer Sciences, vol. 2817, Springer Verlag, September 2003, p. 164-178.

[4] C. DEMAREY, G. HARBONNIER, R. ROUVOY, P. MERLE. *Benchmarking the Round-Trip Latency of Various Java-Based Middleware Platforms*, in "Studia Informatica Universalis Regular Issue", ISBN: 2-912590-31-0, vol. 4, n° 1, May 2005, p. 7-24.

[5] A. FLISSI, C. GRANSART, P. MERLE. *A Component-based Software Infrastructure for Ubiquitous Computing*, in "Proceedings of the 4th International Symposium on Parallel and Distributed Computing (ISPDC 2005), Lille, France", ISBN: 0-7695-2434-6, IEEE, July 2005, p. 183-190.

[6] R. MARVIE. *Vers des patrons de méta-modélisation*, in "Interopérabilité des systèmes distribués: des modèles aux intergiciels, Special NUMBER of Technique et Science Informatique (TSI)", vol. 23, n° 10, 2004, p. 1355-1382.

[7] A. MULLER, O. CARON, B. CARRÉ, G. VANWORMHOUDT. *On some properties of Parameterized Model Applications*, in "European Conference on Model Driven Architecture (ECMDA)", November 2005.

[8] R. PAWLAK, J.-P. RETAILLÉ, L. SEINTURIER. *Foundations of AOP for J2EE Development*, ISBN: 1-59059-507-6, APress, September 2005.

[9] R. PAWLAK, L. SEINTURIER, L. DUCHIEN, G. FLORIN, F. LEGOND-AUBRY, L. MARTELLI. *JAC : An Aspect-based Distributed Dynamic Framework*, in "Software Practise and Experience (SPE)", vol. 34, n° 12, October 2004, p. 1119-1148.

[10] R. ROUVOY, P. MERLE. *Abstraction of Transaction Demarcation in Component-Oriented Platforms*, in "Proceedings of the fourth ACM/IFIP/USENIX International Middleware Conference, Middleware 2003, Rio de Janeiro, Brazil", Lecture Notes in Computer Science, ISBN: 3-540-40317-5, vol. 2972, Springer Verlag, June 2003, p. 305-323.

## Doctoral dissertations and Habilitation theses

[11] O. BARAIS. *Construire et Maîtriser l'évolution d'une architecture logicielle à base de composants*, Ph. D. Thesis, Laboratoire d'Informatique Fondamentale de Lille, Lille, France, November 2005.

[12] S. BONNET. *Une démarche dirigée par les modèles pour la personnalisation des applications embarquées dans les cartes à puces*, Ph. D. Thesis, Laboratoire d'Informatique Fondamentale de Lille, Lille, France, March 2005.

[13] L. SEINTURIER. *Réflexivité, aspects et composants pour l'ingénierie des intergiciels et des applications réparties*, Ph. D. Thesis, Habilitation à diriger des Recherches, Laboratoire d'Informatique de l'Université Pierre et Marie Curie, Paris, France, December 2005.

[14] P. TESSIER. *Conception de modèles de familles de systèmes temps réel*, Ph. D. Thesis, Laboratoire d'Informatique Fondamentale de Lille, Lille, France, December 2005.

## Articles in refereed journals and book chapters

[15] O. BARAIS, A. MULLER, N. PESSEMIER. *Extension de Fractal pour le support des vues au sein d'une architecture logicielle*, in "L'objet, OCM-SI", to appear, Hermes, 2005.

[16] A. BEUGNARD, O. CARON, J. THIBAULT, B. TRAVERSON. *Assemblage de composants par contrats*, in "L'objet, OCM-SI", to appear, Hermes, 2005.

[17] O. CARON, B. CARRÉ, A. MULLER, G. VANWORMHOUDT. *Mise en oeuvre d'aspects fonctionnels réutilisables par adaptation*, in "L'objet, Programmation par Aspects", vol. 11, n° 3, Hermes, jan 2005, p. 105-118.

[18] K. GYBELS, R. WUYTS, S. DUCASSE, M. D'HONDT. *Inter-Language Reflection: A Conceptual Model and Its Implementation*, in "Elsevier JOURNAL on Computer Languages, Systems and Structures", to appear, vol. 26, 2005.

[19] R. MARVIE, L. DUCHIEN, M. BLAY-FORNARINO. *Les plates-formes d'exécution et l'IDM*, ISBN: 2-7462-1213-7, Hermès Science, 2005.

[20] R. MARVIE. *Langages de description d'architectures: un état de l'art*, to appear, Hermès Sciences, 2005.

## Publications in Conferences and Workshops

[21] O. BARAIS, J. LAWALL, A.-F. L. MEUR, L. DUCHIEN. *Providing Support for Safe Software Architecture Transformations*, in "Working Session of the 5th Working IEEE/IFIP Conference on Software Architecture (WICSA 2005), Pittsburg, PA, USA", to appear, nov 2005.

[22] M. A. CIBRAN, M. D'HONDT, V. JONCKERS. *Mapping high-level business rules to and through aspects*, in "2ème Journée Francophone sur le Développement de Logiciels par Aspects (JFDLPA 2005), Lille, France", sep 2005.

[23] D. DIAZ, L. SEINTURIER, L. DUCHIEN, P. FLAMENT. *Un modèle pour la séparation et la traçabilité des préoccupations*, in "Atelier Evolution du Logiciel à LMO'05, Bern, Swiss", March 2005.

[24] A. FLISSI, C. GRANSART, P. MERLE. *A component-based software infrastructure for contextual transportation applications*, in "The 5th International Conference on Intelligent Transportation System - Telecommunication (ITS-T 2005), Brest, France", June 2005.

[25] A. FLISSI, C. GRANSART, P. MERLE. *A service discovery and automatic deployment component-based software infrastructure for Ubiquitous Computing*, in "Ubiquitous Mobile Information and Collaboration Systems, CAiSE Workshop (UMICS 2005), Porto, Portugal", June 2005.

[26] A. FLISSI, C. GRANSART, P. MERLE. *Une infrastructure composants pour des applications ubiquitaires*, in "2nde conférence Ubiquité et Mobilité (UbiMob 2005), Grenoble, France", June 2005.

[27] A. FLISSI, P. MERLE. *Une démarche dirigée par les modèles pour construire les machines de déploiement des intergiciels à composants*, in "Langages et Modèles à Objets (LMO 2005), Bern, Swiss", Hermès Sciences, Mars 2005.

[28] T. HANH-MISSI, L. DUCHIEN, P. BEDU, J. PERRIN, H.-Q. NGUYEN. *Figures de Transformation pour des Architectures Logicielles*, in "Langages et Modèles à Objets, LMO 2005, Bern, Swiss", L'objet, Hermès Sciences, March 2005.

[29] J. LAWALL, H. DUCHESNE, G. MULLER, A.-F. L. MEUR. *Bossa Nova: Introducing Modularity into the Bossa Domain-Specific Language*, in "Fourth International Conference on Generative Programming and Component Engineering (GPCE 2005), Tallinn, Estonia", to appear, sep 2005.

[30] F. LOIRET, D. SERVAT. *Insights on real time systems architecture modelling from a software engineering viewpoint*, in "Work-in-Progress Session of the 17th Euromicro Conference on Real-Time Systems (ECRTS 2005), Palma de Mallorca, Balearic Islands, Spain", to appear, July 2005.

[31] R. MARVIE. *Picolo: A Simple Python Framework for Introducing Component Principles*, in "Euro Python Conference 2005, Goteborg, Sweden", jun 2005.

[32] R. MARVIE, M. NEBUT. *Un cadre de travail pour l'évolution contrôlée des modèles du logiciel*, in "Premier atelier de travail sur l'évolution du logiciel, Berne, Suisse", mar 2005.

[33] W. D. MEUTER, T. D'HONDT, J. DEDECKER, S. MOSTINCKX, T. V. CUTSEM. *First-Class Methods: A Fresh Look at Marrying OOP and FP*, in "Workshop on Multiparadigm Programming With OO Languages at OOPSLA'05", oct 2005.

[34] W. D. MEUTER, E. TANTER, S. MOSTINCKX, T. V. CUSTEM, J. DEDECKER. *Flexible Object Encapsulation for Ambient-Oriented Programming*, in "Dynamic Languages Symposium (DLS) at OOPSLA'05", oct 2005.

[35] E. V. PAESSCHEN, W. D. MEUTER, M. D'HONDT. *Role Modeling in SelfSync With Warped Hierarchies*, in "AAAI Fall Symposium on Roles, Arlington, Virginia, USA", to appear, nov 2005.

[36] E. V. PAESSCHEN, W. D. MEUTER, M. D'HONDT. *SelfSync: A Dynamic Round-Trip Engineering Environment*, in "ACM/IEEE 8th International Conference on Model-Driven Engineering Languages and Systems (MoDELS'05 - formerly UML conference series), Montego Bay, Jamaica", to appear, oct 2005.

[37] R. PAWLAK, L. DUCHIEN, L. SEINTURIER. *Ensuring Safe Around Advice Composition*, in "Proceedings of the 7th IFIP International Conference on Formal Methods for Open Object-based Distributed Systems (FMOODS 2005), Athens, Greece", June 2005.

[38] R. PAWLAK. *Spoon : Annotation-Driven Program Transformation - The AOP Case*, in "Proceedings of the First workshop on Aspect-Oriented Middleware Development (AOMD 2005), ACM/IFIP/USENIX 6th International Middleware Conference, Grenoble, France", November 2005.

[39] N. PESSEMIER, O. BARAIS, L. SEINTURIER, T. COUPAYE, L. DUCHIEN. *A Three Level Framework for Adapting Component Based Architectures*, in "2nd Workshop on Coordination and Adaptation Techniques for Software Entities (WCAT) at ECOOP'05", jul 2005.

[40] P. SERRANO-ALVARADO, R. ROUVOY, P. MERLE. *Self-Adaptive Component-Based Transaction Commit Management*, in "Proceedings of the 4th Workshop on Adaptive and Reflective Middleware (ARM 2005), Grenoble, France", ACM Press, November 2005.

[41] T. SOUED, N. YAHIAOUI, L. SEINTURIER, B. TRAVERSON. *Techniques d'aspect pour la gestion de la mémoire répartie dans un environnement CORBA/C++*, in "5ème Colloque International sur les Nouvelles Technologies de la Répartition (NOTERE'05)", sep 2005.

[42] P. TESSIER, S. GÉRARD, F. TERRIER, J.-M. GEIB. *Using variation propagation for Model-Driven Management of a System Family*, in "Sofware Product Line Conference (SPLC), Rennes", Springer's LNCS, 2005.

[43] G. VANWORMHOUDT. *Précision et Vérification de Métamodèles avec EMF et OCL*, in "Actes des journée Objects Composants et Modèle (OCM) du GDR ALP, Bern, Swiss", nov 2005, p. 19–28.

## Miscellaneous

[44] E. DAGHFAL. *Modélisation structurée des systèmes*, Technical report, Laboratoire d'Informatique Fondamentale de Lille (LIFL), Lille, France, jun 2005.

[45] J. DUBUS. *Modélisation et génération automatique d'infrastructures de déploiement d'applications réparties*, Technical report, Laboratoire d'Informatique Fondamentale de Lille (LIFL), Lille, France, June 2005.

[46] A. SAMSENESENA. *Langage générique pour la séparation et la composition des préoccupations*, Technical report, Laboratoire d'Informatique Fondamentale de Lille (LIFL), Lille, France, jun 2005.

## Bibliography in notes

[47] J. L. PFALTZ, M. NAGL, B. BÖHLEN (editors). *Applications of Graph Transformations with Industrial Relevance, Second International Workshop, AGTIVE 2003, Charlottesville, VA, USA, September 27 - October 1, 2003, Revised Selected and Invited Papers*, Lecture Notes in Computer Science, vol. 3062, Springer, 2004.

[48] J. ARAÚJO, A. RASHID, B. TEKINERDOGAN, A. MOREIRA, P. CLEMENTS (editors). *Early Aspects 2003: Aspect-Oriented Requirements Engineering and Architecture Design*, March 2003, http://www.cs.bilkent.edu.tr/AOSD-EarlyAspects/.

[49] J. ALDRICH, C. CHAMBERS, D. NOTKIN. *ArchJava: connecting software architecture to implementation*, in "Proceedings of the 24th International Conference on Software Engineering (ICSE-02), New York", ACM Press, May 19–25 2002, p. 187–197.

[50] AS-2 EMBEDDED COMPUTING SYSTEMS COMMITTEE SAE. *Architecture Analysis & Design Language (AADL)*, November 2004, SAE Standards n o AS5506.

[51] O. BARAIS, E. CARIOU, L. DUCHIEN, N. PESSEMIER, L. SEINTURIER. *TranSAT: A Framework for the specification of Software Architecture Evolution*, in "ECOOP First International Workshop on Coordination and Adaptation Techniques for Software Entities - WCAT04, Oslo - Norway", June 2004.

[52] O. BARAIS, L. DUCHIEN. *SafArchie : Maîtriser l'Evolution d'une Architecture Logicielle*, in "Langages, Modèles et Objets - Journées Composants - LMO 2004-JC 2004, Lille - France", L'objet, vol. 10, n° 2-3/2004, Hermès Sciences, March 2004, p. 103-116.

[53] O. BARAIS, L. DUCHIEN. *SafArchie Studio: An ArgoUML extension to build Safe Architectures*, in "Workshop on Architecture Description Languages - WADL 2004, Toulouse - France", August 2004.

[54] O. BARAIS, L. DUCHIEN, L. SEINTURIER. *SafArchie ADL : Construire et Déployer une Architecture Logicielle Typée*, Research Report LIFL, n° 2004-n10, Laboratoire d'Informatique Fondamentale de Lille, September 2004.

[55] A. BEUGNARD, J.-M. JÉZÉQUEL, N. PLOUZEAU, D. WATKINS. *Making Components Contract Aware*, in "Computer", vol. 32, n° 7, July 1999, p. 38–45.

[56] P. BROWNE. *Using Drools in Your Enterprise Java Application*, in "OnJava", n° 24-08, 2005.

[57] E. BRUNETON, T. COUPAYE, J. STEFANI. *The Fractal Component Model, version 2.0-3*, Online documentation, February 2004, http://fractal.objectweb.org/specification/.

[58] F. BUENO, D. CABEZA, M. CARRO, M. HERMENEGILDO, P. LÓPEZ-GARCÍA, G. PUEBLA. *The Ciao Prolog system. Reference manual*, Technical report, n° CLIP3/97.1, School of Computer Science, Technical University of Madrid (UPM), August 1997, http://www.clip.dia.fi.upm.es/.

[59] O. CARON, B. CARR, A. MULLER, G. VANWORMHOUDT. *A Framework for Supporting Views in Component Oriented Information Systems*, in "International Conference on Object-Oriented Information Systems, Geneva - Switzerland", Lecture Notes in Computer Sciences, vol. 2817, Springer Verlag, September 2003, p. 164-178.

[60] O. CARON, B. CARRÉ, L. DEBRAUWER. *An Original View Mechanism for the CORBA middleware*, in "TOOLS Europe 2000, Le Mont Saint Michel, France", IEEE Computer Society, June 2000.

[61] O. CARON, B. CARRÉ, L. DEBRAUWER. *Contextualization of OODB Schemas in CROME*, september 2000.

[62] O. CARON, B. CARRÉ, A. MULLER, G. VANWORMHOUDT. *Formulation of UML 2 Template Binding in OCL*, in "7th International Conference on UML Modeling Groups and Applications - UML 2004, Lisbon - Portugal", October 2004.

[63] O. CARON, B. CARRÉ, A. MULLER, G. VANWORMHOUDT. *Mise en oeuvre d'aspects fonctionnels réutilisables par adaptation*, in "Première journée Francophone sur le Développement de Logiciels par Aspects - JFDLPA 2004, Paris - France", September 2004.

[64] S. CLARKE. *Extending standard UML with model composition semantics*, in "Science of Computer Programming, Elsevier Science", 2002.

[65] P. COLLET, R. ROUSSEAU, T. COUPAYE, N. RIVIERRE. *A Contracting System for Hierarchical Components.*, in "CBSE", 2005, p. 187-202.

[66] T. COUPAYE, E. BRUNETON, J.-B. STÉFANI. *The ObjectWeb Fractal Specification*, 2002, http://fractal.objectweb.org.

[67] D. D'SOUZA, A. WILLS. *Objects, Components and Frameworks With UML: The Catalysis Approach*, Addison-Wesley, 1999.

[68] L. DEBRAUWER. *Des vues aux contextes pour la structuration fonctionnelle de bases de données à objets en CROME*, Ph. D. Thesis, Laboratoire d'Informatique Fondamentale de Lille I, Lille, décembre 1998.

[69] R. FILMAN, D. FRIEDMAN. *Aspect-Oriented Programming is Quantification and Obliviousness*, October 2000, http://citeseer.ist.psu.edu/filman00aspectoriented.html.

[70] E. GAMMA, R. HELM, R. JOHNSON, J. VLISSIDES, G. BOOCH. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Westley Professional Computing, USA, 1995.

[71] J.-M. GEIB, C. GRANSART, P. MERLE. *CORBA : des concepts la pratique*, Masson, 1997.

[72] R. HAYTON. *FlexiNet Open ORB Framework*, Technical report, APM ltd, Poseidon House, Castle Park, Cambridge, UK, 1997.

[73] INRIA. *The ObjectWeb Home Page*, 2003, http://www.objectweb.org.

[74] V. ISSARNY, C. KLOUKINAS, A. ZARRAS. *Systematic Aid for Developing Middleware Architectures*, in "Communications of the ACM", vol. 45, n° 6, June 2002, p. 53 - 58.

[75] V. ISSARNY, T. SARIDAKIS, A. ZARROZ. *A Survey of Architecture Definition Languages*, Technical report, C3DS Project, June 1998.

[76] S. KENT. *Model Driven Engineering*, in "Proceedings of IFM 2002", LNCS 2335, Springer-Verlag, May 2002, p. 286-298, http://www.cs.kent.ac.uk/pubs/2002/1594.

[77] G. KICZALES, J. LAMPING, A. MENDHEKAR, C. MAEDA, C. LOPES, J. LOINGTIER, J. IRWIN. *Aspect-Oriented Programming*, in "Proceedings of the 11th European Conference on Object-Oriented Programming (ECOOP'97)", Lecture Notes in Computer Science, vol. 1241, Springer, juin 1997, p. 220-242.

[78] F. KON, F. COSTA, G. BLAIR, R. H. CAMPBELL. *The Case for Reflexive Middleware*, in "Communications of the ACM", vol. 45, n° 6, juin 2002, p. 33 - 38.

[79] B. KOSAYBA, P. MERLE, R. MARVIE, J.-M. GEIB. *Production d'environnements graphiques à partir de méta-modèles*, in "Journée de travail du groupe OCM (GDR ALP)", Laboratoire d'Informatique Fondamentale de Lille, February 2003.

[80] LIFL, OMG. *CORBA Scripting Language Specification, version 1.0*, OMG TC Document formal/2001-06-05, Technical report, juin 2001.

[81] S. LEBLANC, R. MARVIE, P. MERLE, J.-M. GEIB. *Les intergiciels, développements récents dans CORBA, JavaRMI et les agents mobiles*, Chapter : TORBA : contrats de courtage pour CORBA ISBN: 2-7462-0432-0, Hermès Sciences, April 2002, p. 47-72.

[82] T. LEDOUX. *OpenCORBA: a Reflexive Open Broker*, in "Proceedings of the 2nd International Conference Reflexion'99, Saint-Malo, France", LECTURE NOTES IN COMPUTER SCIENCE (editor). , vol. 1616, Springer-Verlag, juillet 1999.

[83] T. LEDOUX, ET AL.. *Etat de l'art sur l'adaptabilité*, Technical report, n° D1.1, Projet RNTL Arcad, December 2001.

[84] R. MARVIE. *Séparation des préoccupations et méta-modélisation pour environnements de manipulation d'architectures logicielles à base de composants*, Ph. D. Thesis, Laboratoire d'Informatique Fondamentale de Lille, December 2002.

[85] R. MARVIE, P. MERLE, J.-M. GEIB. *A Dynamic Platform for CORBA Component Based Applications*, in "First International Conference on Software Engineering Applied to Networking and Parallel/ Distributed Computing (SNPD'00)", ISBN : 0-9700776-0-2, May 2000.

[86] R. MARVIE, P. MERLE, J.-M. GEIB. *Towards a Dynamic CORBA Component Platform*, in "Proceedings of the 2nd International Symposiumon Distributed Object Applications (DOA'2000), Dallas, Texas, USA", ISBN : 0-7695-0819-7, IEEE, September 2000, p. 305-314.

[87] R. MARVIE, P. MERLE, J.-M. GEIB, M. VADET. *OpenCCM : une plate-forme ouverte pour composants CORBA*, in "Actes de la 2ème Conférence Française sur les Systèmes d'Exploitation (CFSE'2)", April 2001, p. 1-12.

[88] N. MEDVIDOVIC, R. N. TAYLOR. *A Classification and Comparison Framework for Software Architecture Description Languages*, in "IEEE Transactions on Software Engineering", vol. 26, n° 1, janvier 2000, 23.

[89] P. MERLE. *CorbaScript - CorbaWeb : propositions pour l'accès à des objets et services répartis*, Ph. D. Thesis, Laboratoire d'Informatique Fondamentale de Lille, Lille, 1997.

[90] M. MEZINI, K. OSTERMANN. *Conquering aspects with Caesar*, in "AOSD '03: Proceedings of the 2nd international conference on Aspect-oriented software development, New York, NY, USA", ISBN: 1-58113-660-9, ACM Press, 2003, p. 90–99.

[91] T.-A. MISSI, P. BEDU, L. DUCHIEN, H. NGUYEN, J. PERRIN. *Toward Structural and Behavioral Analysis for Component Models*, in "The FSE 2004 Workshop on Specification and Verification of Component-Based Systems - SAVCBS 2004, Newport Beach - CA - USA", November 2004, p. 138-141.

[92] A. MULLER, O. CARON, B. CARRÉ, G. VANWORMHOUDT. *Réutilisation d'aspects fonctionnels : des vues aux composants*, in "Proceedings of LMO 03", Hermès Sciences, January 2003, p. 241-255.

[93] A. MULLER. *Reusing functional aspects : from composition to parameterization*, in "Aspect-Oriented Modeling Workshop - AOM 2004, Lisbon - Portugal", October 2004.

[94] OMG. *OMG Model-Driven Architecture Home Page*, http://www.omg.org/mda.

[95] OMG. *OMG UML Home Page*, http://www.uml.org.

[96] OMG. *Common Object Broker Architecture Specification, Version 3.0*, OMG TC Document formal/2002-06-01, Technical report, juin 2002.

[97] R. PAWLAK. *La programmation orientée aspect interactionnelle pour la constructions d'applications à préoccupations multiples*, Ph. D. Thesis, Conservatoire National des Arts et Métiers, Paris, december 2002.

[98] R. PAWLAK, J. RETAILLÉ, L. SEINTURIER. *La programmation orientée aspect pour Java/J2EE*, 2-212-11408-7, Eyrolles, 2004.

[99] R. PAWLAK, L. SEINTURIER, L. DUCHIEN, G. FLORIN, F. LEGOND-AUBRY, L. MARTELLI. *JAC : An Aspect-based Distributed Dynamic Framework*, in "Software Practise and Experience (SPE)", vol. 34, n° 12, October 2004, p. 1119-1148.

[100] R. PAWLAK, L. SEINTURIER, L. DUCHIEN, L. MARTELLI, F. LEGOND-AUBRY, G. FLORIN. *Aspect-Oriented Software Development with Java Aspect Components*, in "Aspect-Oriented Software Development (AOSD)", S. CLARKE, B. FILMAN, T. ELRAD, M. AKSIT (editors). , 0-321-21976-7, Addison-Wesley, September 2004.

[101] J. D. POOLE. *Model-Driven Architecture: Vision, Standards And Emerging Technologies*, in "ECOOP 2001, Workshop on Metamodeling and Adaptive Object Models", April 2001.

[102] ROBERT FRANCE AND GERI GEORG AND INDRAKSHI RAY. *Supporting Multi-Dimensional Separation of Design Concerns*, in "AOSD Workshop on AOM: Aspect-Oriented Modeling with UML", march 2003.

[103] R. ROUVOY, P. MERLE. *GoTM: An Open Framework for Building Heterogeneous Transaction Services*, submission, Technical report, INRIA Futurs, Lille, France, december 2005.

[104] K. SAIKOSKI, G. COULSON, G. BLAIR. *Configurable and Reconfigurable Group Services in a Component Based Middleware Environment*, Distributed Multimedia Research Group, Department of Computing, Lancaster University, 2001.

[105] D. SCHWEISGUTH. *Second-generation aspect-oriented programming*, in "JavaWorld", July 2004, http://www.javaworld.com/javaworld/jw-07-2004/jw-0705-aop.html.

[106] G. STRAW, G. GEORG, E. SONG, S. GHOSH, R. FRANCE, J. M. BIEMAN. *Model Composition Directives*, in "UML 2004 - The Unified Modeling Language. Model Languages and Applications. 7th International Conference, Lisbon, Portugal, October 11-15, 2004, Proceedings", T. BAAR, A. STROHMEIER, A. MOREIRA, S. J. MELLOR (editors). , LNCS, vol. 3273, Springer, 2004, p. 84–97.

[107] D. SUVÉE, W. VANDERPERREN, V. JONCKERS. *JAsCo: an aspect-oriented approach tailored for component based software development*, in "AOSD '03: Proceedings of the 2nd international conference on Aspect-oriented software development, New York, NY, USA", ISBN: 1-58113-660-9, ACM Press, 2003, p. 21–29.

[108] G. TAENTZER. *AGG: A Graph Transformation Environment for Modeling and Validation of Software.*, in "AGTIVE", 2003, p. 446-453.

[109] TRISKELL. *KerMeta Home Page*, http://www.kermeta.org.

[110] G. VANWORMHOUDT. *CROME : un cadre de programmation par objets structurés en contextes*, Ph. D. Thesis, Laboratoire d'Informatique Fondamentale de Lille I, Lille, 1999.

[111] J. WARMER, A. KLEPPE. *The Object Constraint Language – Second Edition, Getting Your Models Ready for MDA*, Addison-Wesley, 2003.