# OBASCO
# OBjects, ASpects, and COmponents

Head: Pierre Cointe

École des Mines de Nantes
Department of Computer Science – OBASCO Group
INRIA Research Centre Rennes - Bretagne Atlantique – LINA

09-2007 / Prague / Econet

# Outline

1. History of OBASCO
2. Current Composition
3. Objectives
4. Strategy
5. Results

# Outline

1. History of OBASCO
2. Current Composition
3. Objectives
4. Strategy
5. Results

# Outline

1. History of OBASCO
2. Current Composition
3. Objectives
4. Strategy
5. Results

# Outline

1. History of OBASCO
2. Current Composition
3. Objectives
4. Strategy
5. Results

# Outline

1. History of OBASCO
2. Current Composition
3. Objectives
4. Strategy
5. Results

# History

- 1992: Creation of a group at EMN on object-oriented programming
  - D. Badouel, I. Borne, A. Réquilé and J. Malenfant
- 1995/1999: Common laboratory with OTI/IBM (Jules Verne) about programming environments for Smalltalk and Java
  - P. Mulet, P. Krief, F. Rivard, and Y-G. Guéhéneuc
- 2002: OBASCO creation as a new INRIA project associated to UR Rennes
- 2003: OBASCO got the label “INRIA project”

# History

- 1992: Creation of a group at EMN on object-oriented programming
  - D. Badouel, I. Borne, A. Réquilé and J. Malenfant
- 1995/1999: Common laboratory with OTI/IBM (Jules Verne) about programming environments for Smalltalk and Java
  - P. Mulet, P. Krief, F. Rivard, and Y-G. Guéhéneuc
- 2002: OBASCO creation as a new INRIA project associated to UR Rennes
- 2003: OBASCO got the label “INRIA project”

# History

- 1992: Creation of a group at EMN on object-oriented programming
  - D. Badouel, I. Borne, A. Réquilé and J. Malenfant
- 1995/1999: Common laboratory with OTI/IBM (Jules Verne) about programming environments for Smalltalk and Java
  - P. Mulet, P. Krief, F. Rivard, and Y-G. Guéhéneuc
- 2002: OBASCO creation as a new INRIA project associated to UR Rennes
- 2003: OBASCO got the label “INRIA project”

# History

- 1992: Creation of a group at EMN on object-oriented programming
  - D. Badouel, I. Borne, A. Réquilé and J. Malenfant
- 1995/1999: Common laboratory with OTI/IBM (Jules Verne) about programming environments for Smalltalk and Java
  - P. Mulet, P. Krief, F. Rivard, and Y-G. Guéhéneuc
- 2002: OBASCO creation as a new INRIA project associated to UR Rennes
- 2003: OBASCO got the label “INRIA project”

# History

- 1992: Creation of a group at EMN on object-oriented programming
  - D. Badouel, I. Borne, A. Réquilé and J. Malenfant
- 1995/1999: Common laboratory with OTI/IBM (Jules Verne) about programming environments for Smalltalk and Java
  - P. Mulet, P. Krief, F. Rivard, and Y-G. Guéhéneuc
- 2002: OBASCO creation as a new INRIA project associated to UR Rennes
- 2003: OBASCO got the label “INRIA project”

# History

- 1992: Creation of a group at EMN on object-oriented programming
  - D. Badouel, I. Borne, A. Réquilé and J. Malenfant
- 1995/1999: Common laboratory with OTI/IBM (Jules Verne) about programming environments for Smalltalk and Java
  - P. Mulet, P. Krief, F. Rivard, and Y-G. Guéhéneuc
- 2002: OBASCO creation as a new INRIA project associated to UR Rennes
- 2003: OBASCO got the label “INRIA project”

# Members

- **8 permanent staffs**
  P. Cointe (PR 92)
  R. Douence (MA 98)
  T. Ledoux (MA 98)
  J-M. Menaud (MA 00)
  G. Muller (PR 02)
  J. Noyé (MA 96)
  M. Südholt (MA 97)
  J-C. Royer (PR 02)
- **1 associated staff**
  H. Grall
- **1 regular invited**
  J. Lawall (Diku)
- **2 post PhDs**
  P-C. David (Nantes)
  J. Noppen (Twente)
- **18 PhD Students**
  A. Assaf
  F. Baligand
  S. Djoko
  F. Hermenier
  M. Leger
  F. Minjat
  H. Nguyen
  R. Urunela
  J. Berniolles
  C. Augier
  L-D. Bena
  F. Fernand
  H. Arboled
  N. Loriant
  A. Nunez
  S. Pavel
  K. Garces
  C. Tavares

# Members

- **8 permanent staffs**
  P. Cointe (PR 92)
  R. Douence (MA 98)
  T. Ledoux (MA 98)
  J-M. Menaud (MA 00)
  G. Muller (PR 02)
  J. Noyé (MA 96)
  M. Südholt (MA 97)
  J-C. Royer (PR 02)
- **1 associated staff**
  H. Grall
- **1 regular invited**
  J. Lawall (Diku)

- **2 post PhDs**
  P-C. David (Nantes)
  J. Noppen (Twente)
- **18 PhD Students**
  A. Assaf
  F. Baligand
  S. Djoko
  F. Hermenier
  M. Leger
  F. Minjat
  H. Nguyen
  R. Urunela
  J. Berniolles
  C. Augier
  L-D. Bena
  F. Fernand
  H. Arboled
  N. Loriant
  A. Nunez
  S. Pavel
  K. Garces
  C. Tavares

# Members

- **8 permanent staffs**
  P. Cointe (PR 92)
  R. Douence (MA 98)
  T. Ledoux (MA 98)
  J-M. Menaud (MA 00)
  G. Muller (PR 02)
  J. Noyé (MA 96)
  M. Südholt (MA 97)
  J-C. Royer (PR 02)
- **1 associated staff**
  H. Grall
- **1 regular invited**
  J. Lawall (Diku)

- **2 post PhDs**
  P-C. David (Nantes)
  J. Noppen (Twente)
- **18 PhD Students**
  A. Assaf
  F. Baligand
  S. Djoko
  F. Hermenier
  M. Leger
  F. Minjat
  H. Nguyen
  R. Urunela
  J. Berniolles
  C. Augier
  L-D. Bena
  F. Fernand
  H. Arboled
  N. Loriant
  A. Nunez
  S. Pavel
  K. Garces
  C. Tavares

# Members

- **8 permanent staffs**
  P. Cointe (PR 92)
  R. Douence (MA 98)
  T. Ledoux (MA 98)
  J-M. Menaud (MA 00)
  G. Muller (PR 02)
  J. Noyé (MA 96)
  M. Südholt (MA 97)
  J-C. Royer (PR 02)
- **1 associated staff**
  H. Grall
- **1 regular invited**
  J. Lawall (Diku)

- **2 post PhDs**
  P-C. David (Nantes)
  J. Noppen (Twente)
- **18 PhD Students**
  A. Assaf
  F. Baligand
  S. Djoko
  F. Hermenier
  M. Leger
  F. Minjat
  H. Nguyen
  R. Urunela
  J. Berniolles
  C. Augier
  L-D. Bena
  F. Fernand
  H. Arboled
  N. Loriant
  A. Nunez
  S. Pavel
  K. Garces
  C. Tavares

# Members

- **8 permanent staffs**
  P. Cointe (PR 92)
  R. Douence (MA 98)
  T. Ledoux (MA 98)
  J-M. Menaud (MA 00)
  G. Muller (PR 02)
  J. Noyé (MA 96)
  M. Südholt (MA 97)
  J-C. Royer (PR 02)
- **1 associated staff**
  H. Grall
- **1 regular invited**
  J. Lawall (Diku)
- **2 post PhDs**
  P-C. David (Nantes)
  J. Noppen (Twente)
- **18 PhD Students**
  A. Assaf
  F. Baligand
  S. Djoko
  F. Hermenier
  M. Leger
  F. Minjat
  H. Nguyen
  R. Urunela
  J. Berniolles
  C. Augier
  L-D. Bena
  F. Fernand
  H. Arboled
  N. Loriant
  A. Nunez
  S. Pavel
  K. Garces
  C. Tavares

# Objectives

- To solve scalability problems in software engineering
- To improve software architectures adaptation
- Two main directions
  - Separation of concern: specific programs for specific problems
  - Correct composition of existing programming artefacts

# Objectives

- To solve scalability problems in software engineering
- To improve software architectures adaptation
- Two main directions
    - Separation of concern: specific programs for specific problems
    - Correct composition of existing programming artefacts

# Objectives

- To solve scalability problems in software engineering
- To improve software architectures adaptation
- Two main directions
  - Separation of concern: specific programs for specific problems
  - Correct composition of existing programming artefacts

# Objectives

- To solve scalability problems in software engineering
- To improve software architectures adaptation
- Two main directions
  - Separation of concern: specific programs for specific problems
  - Correct composition of existing programming artefacts

# Objectives

- To solve scalability problems in software engineering
- To improve software architectures adaptation
- Two main directions
  - Separation of concern: specific programs for specific problems
  - Correct composition of existing programming artefacts

# Building

- Buildings are constructed from several descriptions
- Architecture, Electricity, Heating, Air-Conditioning, Water, Network, ...
- Each of them described with the adequate tools and specific rules
- There are interactions and constraints between them
- To build means to compose them: a really complex process and that must be correct

# Building

- Buildings are constructed from several descriptions
- Architecture, Electricity, Heating, Air-Conditioning, Water, Network, ...
- Each of them described with the adequate tools and specific rules
- There are interactions and constraints between them
- To build means to compose them: a really complex process and that must be correct

# Building

- Buildings are constructed from several descriptions
- Architecture, Electricity, Heating, Air-Conditioning, Water, Network, ...
- Each of them described with the adequate tools and specific rules
- There are interactions and constraints between them
- To build means to compose them: a really complex process and that must be correct

# Building

- Buildings are constructed from several descriptions
- Architecture, Electricity, Heating, Air-Conditioning, Water, Network, ...
- Each of them described with the adequate tools and specific rules
- There are interactions and constraints between them
- To build means to compose them: a really complex process and that must be correct

# Building

- Buildings are constructed from several descriptions
- Architecture, Electricity, Heating, Air-Conditioning, Water, Network, ...
- Each of them described with the adequate tools and specific rules
- There are interactions and constraints between them
- To build means to compose them: a really complex process and that must be correct

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Research Domain

- **Software Engineering**
  - Software components and scalability
  - Programming languages
  - Post object-oriented programming
  - Generative programming
    - Sequential, concurrent and distributed
    - Mechanism for separation and composition
    - Objects versus aspects versus components
  - Model driven engineering: transformation techniques
- ECOOP, OOPSLA, AOSD, GPCE, DOA, PEPM, ASE

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

## Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

## Three Swim Lanes

- Aspect-oriented programming
  - To explicit links between metaobject and aspect
  - To formalize aspect-oriented models
  - To design and implement a language
  - Reverse engineering of legacy code with aspects
- Software component
  - Explicit protocols for components
  - Property verification for components and architectures
  - Understand relations between aspects and components
- Domain specific language
  - Expressiveness, extensibility and compilation
  - Aspect languages, composition and DSL

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Aspects

- Formalization of aspect-oriented programming
  - (C)EAOP: a formal (concurrent) model based on events
  - Static analysis of aspect interactions
  - CASB: Operational semantics for aspect-oriented language
- Aspect-oriented languages
  - Reflex: Reflexive kernel for AOP
  - Arachne: dynamic aspect weaver for C
  - AWED: Distributed aspects
- Reverse engineering
  - Aspects for design patterns in JHotDraw

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Components

- Explicit protocols
  - Compatibility and substitutability for components
  - Component model with symbolic transition systems
  - Verification based on boundedness
  - Runtime and code generation support
- Adaptation
  - Fractal extension
  - Specialization and component generation

# Domain Specific Languages

- Bossa
  - Modularization and aspects for Linux kernel
  - DSL extensibility (hierarchy of schedulers)
  - Verification
- Coccinelle
  - Description and implementation of Linux drivers evolution
- Compilation methods

# Domain Specific Languages

- Bossa
  - Modularization and aspects for Linux kernel
  - DSL extensibility (hierarchy of schedulers)
  - Verification
- Coccinelle
  - Description and implementation of Linux drivers evolution
- Compilation methods

# Domain Specific Languages

- Bossa
  - Modularization and aspects for Linux kernel
  - DSL extensibility (hierarchy of schedulers)
  - Verification
- Coccinelle
  - Description and implementation of Linux drivers evolution
- Compilation methods

# Domain Specific Languages

- Bossa
  - Modularization and aspects for Linux kernel
  - DSL extensibility (hierarchy of schedulers)
  - Verification
- Coccinelle
  - Description and implementation of Linux drivers evolution
- Compilation methods

# Domain Specific Languages

- Bossa
  - Modularization and aspects for Linux kernel
  - DSL extensibility (hierarchy of schedulers)
  - Verification
- Coccinelle
  - Description and implementation of Linux drivers evolution
- Compilation methods

# Domain Specific Languages

- Bossa
  - Modularization and aspects for Linux kernel
  - DSL extensibility (hierarchy of schedulers)
  - Verification
- Coccinelle
  - Description and implementation of Linux drivers evolution
- Compilation methods

# Domain Specific Languages

- Bossa
  - Modularization and aspects for Linux kernel
  - DSL extensibility (hierarchy of schedulers)
  - Verification
- Coccinelle
  - Description and implementation of Linux drivers evolution
- Compilation methods

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects
- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects
- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects
- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects

- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects
- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects
- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects
- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Main Publications

- OOPLSA
  03: UML vs Java
  04: MOP Reflex
- ECOOP
  04: SOM with Reflex
- AOSD
  O3: Cache and aspect
  O4: Aspect intercation
  O5: Arachne
  O6: AWED
- GPCE
  04: Cflow, Bossa/DSL
  05: GP/DSL, Bossa/modules, mutliple aspects
- PEPM
  04: Component specialisation, hierarchy of schedulers
- ASE
  03: Bossa verification, UML binary class
- DOA
  03: asynchronous STS
  04: Boundedness analysis
  06: AWED and web services
- EuroSys
  04: Bossa/component
  06: Linux drivers

# Softwares

- Bossa
- Safran
- Arachne
- AWED/JasCo
- Baton/Stratego/Reflex

# Softwares

- Bossa
- Safran
- Arachne
- AWED/JasCo
- Baton/Stratego/Reflex

# Softwares

- Bossa
- Safran
- Arachne
- AWED/JasCo
- Baton/Stratego/Reflex

# Softwares

- Bossa
- Safran
- Arachne
- AWED/JasCo
- Baton/Stratego/Reflex

## Softwares

- Bossa
- Safran
- Arachne
- AWED/JasCo
- Baton/Stratego/Reflex

# International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

# International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

# International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

# International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

## International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

# International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

# International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

# International Contracts

- REX AOSD Europe 2004-2008
- STREP AMPLE 2006-2009
- IST EasyComp 2001-2003
- Alfa Elastex 2001-2003
- Econet 2007-2008
- Siemens 2006-2007
- Microsoft Research 2002-2004
- IBM Eclipse Fellowships 2002-2004

# National Contracts

- ANR blanc FLFS 2006-2009
- ANR blanc Coccinelle 2005-2008
- ANR/RNTL SADAJ 2006-2009
- ANR/RNTL Selfware 2005-2008
- ACI sécurité CORSS 2003-2006
- ACI sécurité DISPO 2003-2006
- RNTL ARCAD 2000-2004

# National Contracts

- ANR blanc FLFS 2006-2009
- ANR blanc Coccinelle 2005-2008
- ANR/RNTL SADAJ 2006-2009
- ANR/RNTL Selfware 2005-2008
- ACI sécurité CORSS 2003-2006
- ACI sécurité DISPO 2003-2006
- RNTL ARCAD 2000-2004

# National Contracts

- ANR blanc FLFS 2006-2009
- ANR blanc Coccinelle 2005-2008
- ANR/RNTL SADAJ 2006-2009
- ANR/RNTL Selfware 2005-2008
- ACI sécurité CORSS 2003-2006
- ACI sécurité DISPO 2003-2006
- RNTL ARCAD 2000-2004

# National Contracts

- ANR blanc FLFS 2006-2009
- ANR blanc Coccinelle 2005-2008
- ANR/RNTL SADAJ 2006-2009
- ANR/RNTL Selfware 2005-2008
- ACI sécurité CORSS 2003-2006
- ACI sécurité DISPO 2003-2006
- RNTL ARCAD 2000-2004

## National Contracts

- ANR blanc FLFS 2006-2009
- ANR blanc Coccinelle 2005-2008
- ANR/RNTL SADAJ 2006-2009
- ANR/RNTL Selfware 2005-2008
- ACI sécurité CORSS 2003-2006
- ACI sécurité DISPO 2003-2006
- RNTL ARCAD 2000-2004

# National Contracts

- ANR blanc FLFS 2006-2009
- ANR blanc Coccinelle 2005-2008
- ANR/RNTL SADAJ 2006-2009
- ANR/RNTL Selfware 2005-2008
- ACI sécurité CORSS 2003-2006
- ACI sécurité DISPO 2003-2006
- RNTL ARCAD 2000-2004

# National Contracts

- ANR blanc FLFS 2006-2009
- ANR blanc Coccinelle 2005-2008
- ANR/RNTL SADAJ 2006-2009
- ANR/RNTL Selfware 2005-2008
- ACI sécurité CORSS 2003-2006
- ACI sécurité DISPO 2003-2006
- RNTL ARCAD 2000-2004