

# Object-Oriented Connector-Component Architectures

H. Ehrig<sup>1</sup>   B. Braatz<sup>1</sup>   M. Klein<sup>1,2</sup>   F. Orejas<sup>2</sup>  
S. Pérez<sup>2</sup>   E. Pino<sup>2</sup>

<sup>1</sup>Institut für Softwaretechnik und Theoretische Informatik  
Technische Universität Berlin  
Berlin, Germany

<sup>2</sup>Departament de Llenguatges i Sistemes Informàtics  
Universitat Politècnica de Catalunya  
Barcelona, Spain

Formal Foundations of Embedded Software and  
Component-Based Software Architectures  
Edinburgh, 9th April 2005

# Outline

Object-Oriented  
Connector-  
Component  
Architectures

Ehrig, Braatz et al.

## Generic Architecture Framework

Connector-Component Architectures  
Composition by Extension  
Architecture Semantics

Generic  
Architecture  
Framework

Connector-Component  
Architectures

Composition by Extension  
Architecture Semantics

Instantiation to  
UML

Class Diagrams

Sequence Diagrams and  
State Machines

Conclusion

## Instantiation to UML

Class Diagrams  
Sequence Diagrams and State Machines

# Generic Architecture Framework

Object-Oriented  
Connector-  
Component  
Architectures

Ehrig, Braatz et al.

- ▶ Framework for Connector-Component Architectures
- ▶ Generic w. r. t.
  - ▶ Specification / Modeling Techniques
  - ▶ Transformations / Refinements
  - ▶ Embeddings / Inclusions
- ▶ Existing Instantiations (FESCA 2004):
  - ▶ Petri Nets
  - ▶ CCS Process Algebra
- ▶ New Instantiation:
  - ▶ Object-Oriented Specifications / UML Models

Generic  
Architecture  
Framework

Connector-Component  
Architectures

Composition by Extension  
Architecture Semantics

Instantiation to  
UML

Class Diagrams

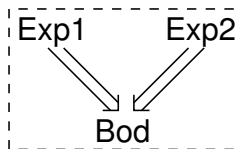
Sequence Diagrams and  
State Machines

Conclusion

# Structure of Components and Connectors

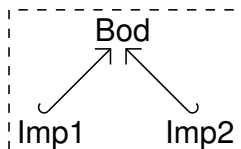
## Component as Computation Unit

- ▶ *Exports* specify Provisions
- ▶ *Body* specifies Realization
- ▶ Connected by *Transformations*



## Connector as Coordination Unit

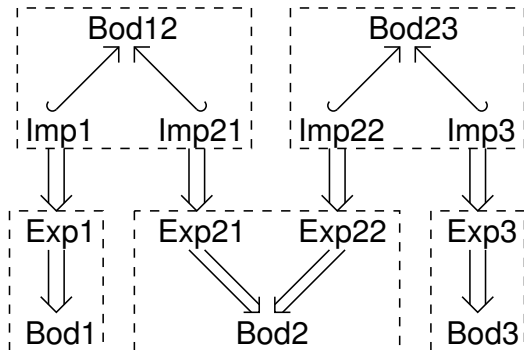
- ▶ *Imports* specify Requirements
- ▶ *Body* specifies Coordination
- ▶ Connected by *Embeddings*



# Connector-Component Architectures

## Architecture Diagrams:

- ▶ Components and Connectors
- ▶ Transformations from Imports to Exports



# Example Package Architecture

Object-Oriented  
Connector-  
Component  
Architectures

Ehrig, Braatz et al.

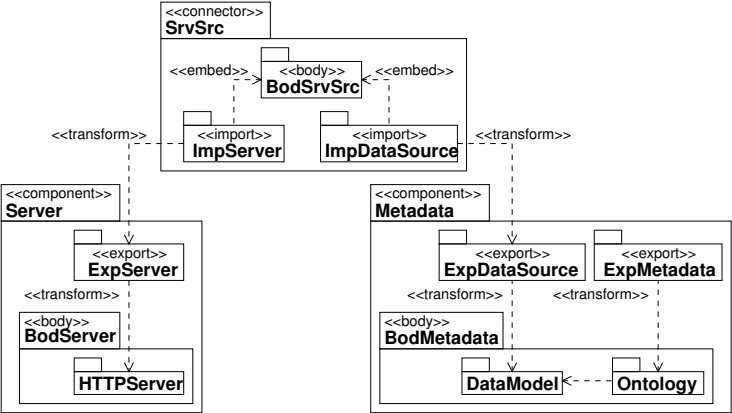
Generic  
Architecture  
Framework

Connector-Component  
Architectures  
Composition by Extension  
Architecture Semantics

Instantiation to  
UML

Class Diagrams  
Sequence Diagrams and  
State Machines

Conclusion



# Parallel Extension Property

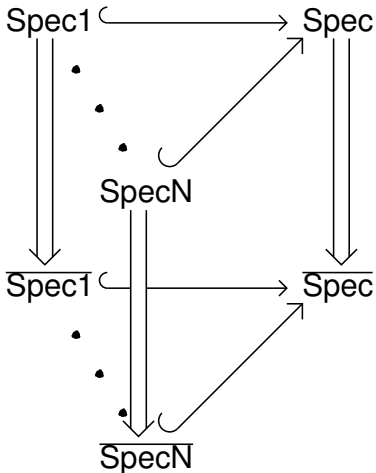
Requirement by the Generic Framework:

Family of Embeddings

and

Family of *Consistent*  
Transformations

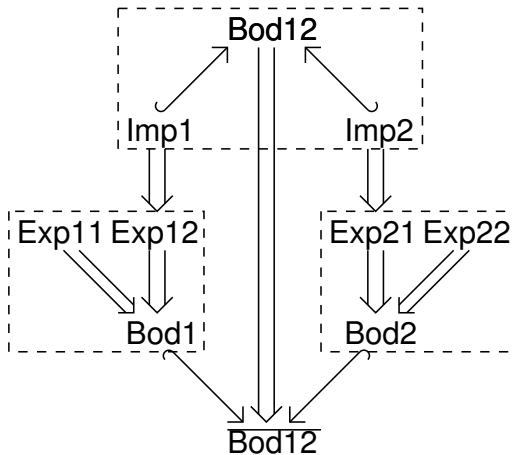
⇒ *Extension* of  
Transformations along  
Embeddings



# Composition of Components

Connector  
combining  
Components

Composed  
Body by Parallel  
Extension



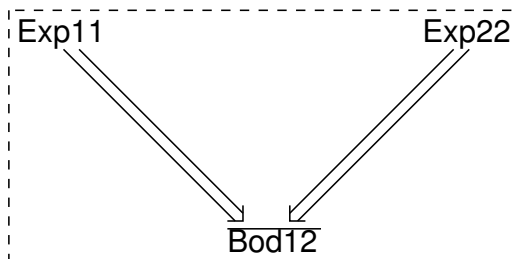
# Composition of Components

Connector  
combining  
Components

Composed  
Body by Parallel  
Extension

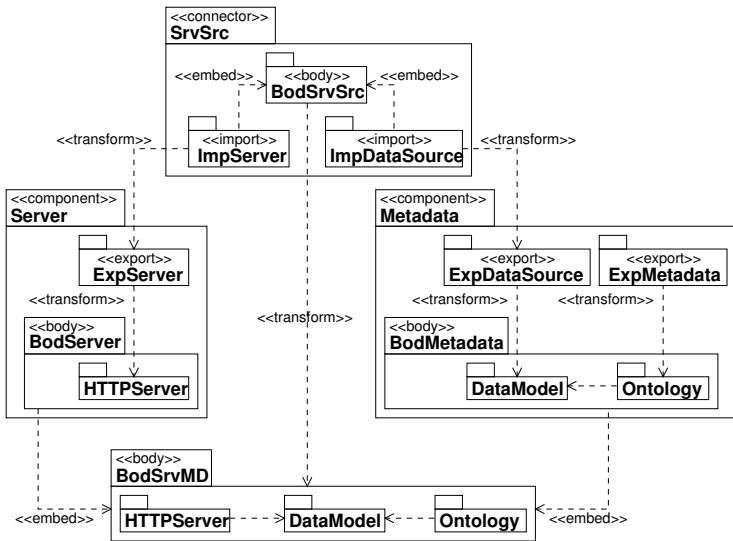
Connected  
Parts Removed

Composition of  
Transformation  
and Embedding



⇒ Single Composed Component

# Composition of the Example



# Composition of the Example

Object-Oriented  
Connector-  
Component  
Architectures

Ehrig, Braatz et al.

Generic  
Architecture  
Framework

Connector-Component  
Architectures

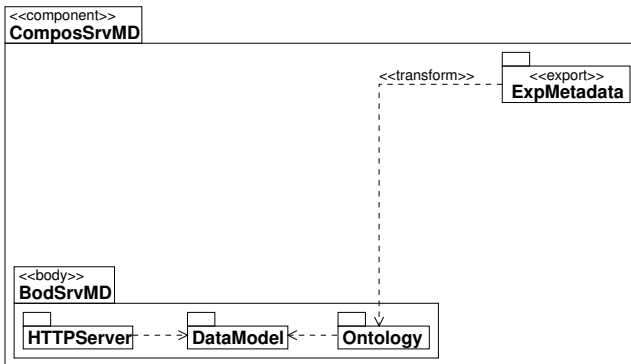
Composition by Extension  
Architecture Semantics

Instantiation to  
UML

Class Diagrams

Sequence Diagrams and  
State Machines

Conclusion



# Architecture Semantics / Main Results

Object-Oriented  
Connector-  
Component  
Architectures

Ehrig, Braatz et al.

Valid for all Instantiations of the Framework  
(Only dependent on Parallel Extension Property)

- ▶ Semantics of Architectures:  
Single Component derived by Iterated Composition
- ▶ Connectors gradually removed by Composition
  - ⇒ Iterated Composition terminates
  - ⇒ Existence of Semantics
- ▶ Compatibility/Associativity of Composition
  - ⇒ Local Confluence of Composition
  - ⇒ Uniqueness of Semantics

Generic  
Architecture  
Framework

Connector-Component  
Architectures  
Composition by Extension  
Architecture Semantics

Instantiation to  
UML

Class Diagrams  
Sequence Diagrams and  
State Machines

Conclusion

# Semantics of Example Architecture

Object-Oriented  
Connector-  
Component  
Architectures

Ehrig, Braatz et al.

Generic  
Architecture  
Framework

Connector-Component  
Architectures

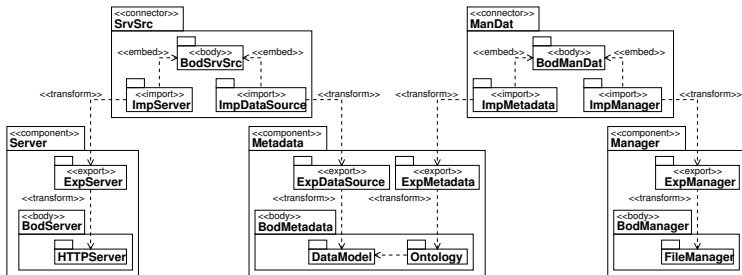
Composition by Extension  
Architecture Semantics

Instantiation to  
UML

Class Diagrams

Sequence Diagrams and  
State Machines

Conclusion



# Semantics of Example Architecture

Object-Oriented  
Connector-  
Component  
Architectures

Ehrig, Braatz et al.

Generic  
Architecture  
Framework

Connector-Component  
Architectures

Composition by Extension

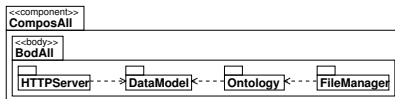
Architecture Semantics

Instantiation to  
UML

Class Diagrams

Sequence Diagrams and  
State Machines

Conclusion

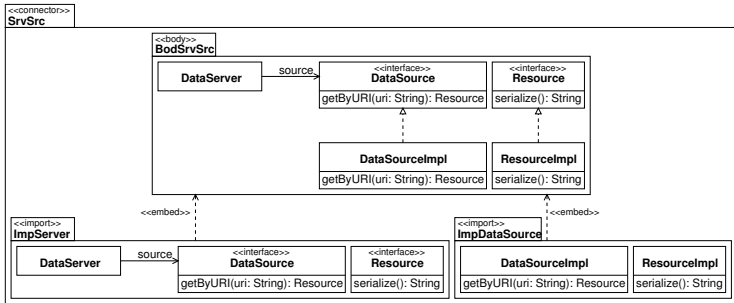


Instantiations of Architecture Framework must provide:

- ▶ Class of Specifications
  - ⇒ Packages containing UML Diagrams
- ▶ Transformations and Embeddings
  - ⇒ Refinements and Inclusions of UML Diagrams
- ▶ Consistency of Transformations and Embeddings
  - ⇒ Parallel Extension Property

# Embeddings of Class Diagrams

- Identification of Classes to be realized in different Components
- No Renamings – just Inclusions



# Transformations of Class Diagrams

- ▶ Realization of Classes by Addition of Methods and Properties
- ▶ Renamings possible (except Interfaces)

Ehrig, Braatz et al.

Generic  
Architecture  
Framework

Connector-Component  
Architectures

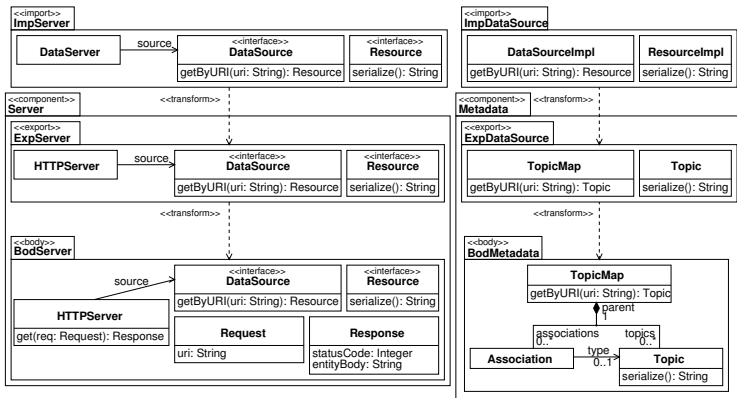
Composition by Extension  
Architecture Semantics

Instantiation to  
UML

Class Diagrams

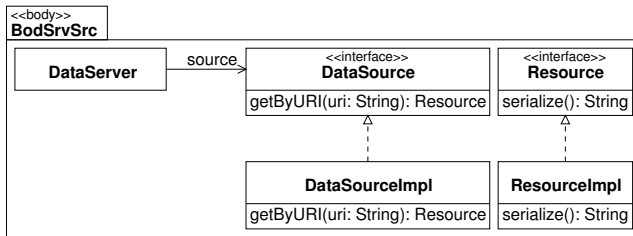
Sequence Diagrams and  
State Machines

Conclusion



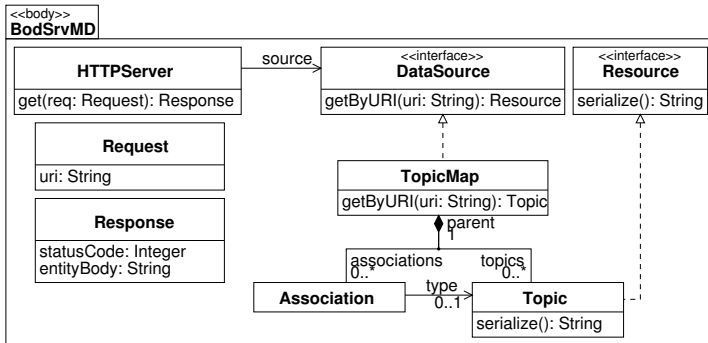
# Extension Property for Class Diagrams

- ▶ Consistency:
  - ▶ Boundary of Embeddings preserved
  - ▶ Common Parts transformed identically
- ▶ Extension by Transformations in larger Context



# Extension Property for Class Diagrams

- ▶ Consistency:
  - ▶ Boundary of Embeddings preserved
  - ▶ Common Parts transformed identically
- ▶ Extension by Transformations in larger Context



# Sequence Diagrams

- ▶ Considering Sets of Scenarios
  - ▶ Embeddings:
    - ▶ Inclusion of each Scenario in the Source into some Scenario of the Target
  - ▶ Transformations:
    - ▶ Translation of Names according to Class Diagram Transformation
    - ▶ Refinement of Lifelines by Sets of Lifelines (Decomposition)
    - ▶ Refinement of Messages by (Sub-)Scenarios
  - ▶ Consistency of Transformations and Embeddings:
    - ▶ Boundary of Embeddings preserved
    - ▶ Common Parts transformed identically
- ⇒ Extension of Sequence Diagram Transformations

# State Machines

- ▶ Considering State Machines assigned to Classes or Methods
  - ▶ Embeddings:
    - ▶ Inclusion of State Machines, such that visible Traces are preserved
  - ▶ Transformations:
    - ▶ Translation of Names according to Class Diagram Transformation
    - ▶ States may be renamed and given Substructure
    - ▶ Possible Traces enhanced
  - ▶ Consistency of Transformations and Embeddings:
    - ▶ Boundary of Embeddings preserved
    - ▶ Common Parts transformed identically
- ⇒ Extension of State Machine Transformations

# Conclusion

## Summary:

- ▶ Theoretical Framework based on Parallel Extension Property
- ▶ New Application of Architecture Framework to UML (FESCA 2004: Instantiations to CCS and Petri Nets)
- ▶ Universal Connectors for Common Architecture Styles

## Future Work:

- ▶ Transformation/Refinement Concept for Further Techniques
- ▶ Application of Graph Transformation to UML

Thank You!