Advances in Architectural Design Rewriting

Alberto Lluch Lafuente

lafuente@di.unipi.it

Department of Computer Science
University of Pisa

Software Engineering for
Service-Oriented Overlay Computers

Sensoria General Meeting
Pisa, ISTI-CNR, 9-11 June 2008
Outline

Overview

Advance 1: ADR4UML4SOA

Advance 2: ADR4SRML

Advance 3: Towards D5a.

Conclusion
Outline

Overview

Advance 1: ADR4UML4SOA

Advance 2: ADR4SRML

Advance 3: Towards D5a.

Conclusion
Architectural Design Rewriting: An Overview

ADR specifications are...

- process algebras interpreted over graphs (designs).
Architectural Design Rewriting: An Overview

ADR specifications are...

- process algebras interpreted over graphs (designs).

- \( W \times W \rightarrow W \)

- \( W \times W \rightarrow W \)
Architectural Design Rewriting: Previous work

Previous work on ADR

- Original presentation [BLM08].
- SRML design and reconfiguration [BLME07].
- Hierarchical graphs, implementation, analysis [BLM08].
- Comparison with logic-based approaches [BBGL08].
- Advertised at R2D2 [BLMT08].
Outline

Overview

Advance 1: ADR4UML4SOA

Advance 2: ADR4SRML

Advance 3: Towards D5a.

Conclusion
ADR4UML4SOA: Idea

What is UML4SOA?
- An UML profile for SOA;
- Extended for style-based design and reconfiguration.
ADR4UML4SOA: Idea

What is UML4SOA?
- An UML profile for SOA;
- Extended for style-based design and reconfiguration.

What is the purpose?
- Give semantics to UML4SOA profile extension.
ADR4UML4SOA: Idea

What is UML4SOA?
- An UML profile for SOA;
- Extended for style-based design and reconfiguration.

What is the purpose?
- Give semantics to UML4SOA profile extension.

How does it work?
- Almost a one-to-one mapping.

More in [BBF+07]
ADR4soa4uml: Configurations

Configurations as Designs

Adances in Architectural Design Rewriting

lafuente@di.unipi.it
ADR4soa4uml: Configurations

Configurations as Designs

- Types $\mapsto$ Types
ADR4soa4uml: Configurations

Configurations as Designs

- Types $\mapsto$ Types
- Subcomps $\mapsto$ Edges
ADR4soa4uml: Configurations

Configurations as Designs
- Types → Types
- Subcomps → Edges
- Ports → Tentacles
ADR4soa4uml: Configurations

Configurations as Designs

- Types ↔ Types
- Subcomps ↔ Edges
- Ports ↔ Tentacles
- Connectors ↔ Edges
ADR4soa4uml: Configurations

Configurations as Designs
- Types \(\leftrightarrow\) Types
- Subcomps \(\leftrightarrow\) Edges
- Ports \(\leftrightarrow\) Tentacles
- Connectors \(\leftrightarrow\) Edges
- Delegates \(\leftrightarrow\) Interface

Overview
- Advance 1: ADR4UML4SOA
- Advance 2: ADR4SRML
- Advance 3: Towards D5a.

Conclusion
**ADR4UML4SOA: Styles**

- Types spread over diagrams
  - Type Graph

E.g. refinable $\mapsto$ non-terminal edges, i.e. variables.

Advances in Architectural Design Rewriting  
lafuente@di.unipi.it
ADR4UML4SOA: Styles

Architectural Styles

- Types spread over diagrams
  \( \rightarrow \) Type Graph
- Productions
  \( \rightarrow \) Productions

Overview

Advance 1: ADR4UML4SOA
Advance 2: ADR4SRML
Advance 3: Towards D5a.
Conclusion
ADR4UML4SOA: Styles

Architectural Styles

- Types spread over diagrams
  - Type Graph
- Productions
  - Productions
- Cardinalities
  - Productions

E.g. sets inductively defined: empty, singleton or union.
ADR4UML4SOA: Reconfigurations

Transformations
ADR4UML4SOA: Reconfigurations

Transformations

- parse(pre) → l
ADR4UML4SOA: Reconfigurations

Transformations
- \( \text{parse(pre)} \rightarrow l \)
- \( \text{parse(post)} \rightarrow r \)
ADR4UML4SOA: Reconfigurations

Transformations
- parse(pre) → l
- parse(post) → r
- l → r
Overview

Advance 1: ADR4UML4SOA

Advance 2: ADR4SRML

Advance 3: Towards D5a.

Conclusion

ADR4UML4SOA: Reconfigurations

Transformations
- parse(pre) → l
- parse(post) → r
- l → r

Similar job for conditional reconfigurations.
Outline

Overview

Advance 1: ADR4UML4SOA

Advance 2: ADR4SRML

Advance 3: Towards D5a.

Conclusion
ADR4SRML: Idea

What is SRML?

▶ Sensoria’s service modelling language.
ADR4SRML: Idea

What is SRML?

- Sensoria’s service modelling language.

What is the purpose?

- Help understand SRML semantics.
- Add information to SRML specifications.
ADR4SRML: Idea

What is SRML?
- Sensoria’s service modelling language.

What is the purpose?
- Help understand SRML semantics.
- Add information to SRML specifications.

How does it work?
- Use previous work [BLME07] to construct the graph.
- Define the behaviour of each single component/connector as edge-to-edge rewrites.
- Synchronise à la SHR.

More in [BBF+07]
ADR4SRML: Past

Read ADR as

- A = well formedness.
- D = diagrams.
- R = composition.

More in [BLM08]
ADR4SRML: Configurations

SRML Diagrams
ADR4SRML: Configurations

SRML Diagrams

Components
ADR4SRML: Configurations

SRML Diagrams

- Components
- Wires
ADR4SRML: Configurations

SRML Diagrams
- Components
- Wires
- Interactions
ADR4SRML: Transitions

**BUSINESS ROLE** Orchestrator is

**ORCHESTRATION**

local \( s: \) [INIT, FAILURE\_DETECTED, CONTEXT\_RECEIVED, GARAGE\&TT, CAR\(+\)GARAGE\&TT, FINAL],

transition StartProcess

  triggeredBy engineFailure
  guardedBy \( s=\) INIT
  effects \( s'=\) FAILURE\_DETECTED
    \( \land \) \( l'=\) currentLocation
  sends askUsrDetails

transition getContextData

  triggeredBy askUsrDetails
  guardedBy \( s=\) FAILURE\_DETECTED
  effects \( s'=\) CONTEXT\_RECEIVED
  sends bookGarage
    \( \land \) bookGarage.collectPnt = \( l \)
**BUSINESS ROLE** Orchestrator is

**ORCHESTRATION**

local s: [INIT,FAILURE_DETECTED,CONTEXT_RECEIVED,
GARAGE&TT,CAR+GARAGE&TT,FINAL],

transition StartProcess
triggeredBy engineFailure
guardedBy s=INIT
effects s'=FAILURE_DETECTED
\wedge l'=currentLocation
sends askUsrDetails

transition getContextData
triggeredBy askUsrDetails
guardedBy s=FAILURE_DETECTED
effects s'=CONTEXT_RECEIVED
sends bookGarage
\wedge \text{bookGarage.collectPnt}=l

Transition

- Lhs = state/guard

![Diagram](image-url)
**BUSINESS ROLE** Orchestrator is

**ORCHESTRATION**

\[
\text{local } s : \{ \text{INIT, FAILURE\_DETECTED, CONTEXT\_RECEIVED, GARAGE\&TT, CAR\&GARAGE\&TT, FINAL} \},
\]

\[
\text{transition } \text{StartProcess}
\]

\[
\text{triggeredBy } \text{engineFailure}
\]

\[
\text{guardedBy } s = \text{INIT}
\]

\[
\text{effects } s' = \text{FAILURE\_DETECTED}
\]

\[
\wedge l' = \text{currentLocation}
\]

\[
\text{sends } \text{askUsrDetails}
\]

\[
\text{transition } \text{getContextData}
\]

\[
\text{triggeredBy } \text{askUsrDetails}
\]

\[
\text{guardedBy } s = \text{FAILURE\_DETECTED}
\]

\[
\text{effects } s' = \text{CONTEXT\_RECEIVED}
\]

\[
\text{sends } \text{bookGarage}
\]

\[
\wedge \text{bookGarage.collectPnt} = l
\]

**Transition**

- Lhs = state/guard
- Label = trigger + sends
**BUSINESS ROLE** Orchestrator is

**ORCHESTRATION**

local $s$: [INIT, FAILURE_DETECTED, CONTEXT_RECEIVED, GARAGE&TT, CAR+GARAGE&TT, FINAL],

transition StartProcess
  triggeredBy engineFailure
  guardedBy $s=$INIT
  effects $s'=$FAILURE_DETECTED
      ∧ $l'=$currentLocation
  sends askUsrDetails

transition getContextData
  triggeredBy askUsrDetails
  guardedBy $s=$FAILURE_DETECTED
  effects $s'=$CONTEXT_RECEIVED
  sends bookGarage
      ∧ bookGarage.collectPnt=$l

**Transition**

- **Lhs** = state/guard
- **Label** = trigger + sends
- **Rhs** = state/effect

(WithContextData)
ADR4SRML: Synchronisation
ADR4SRML: Synchronisation

Synchronisation

- Prem: individual rules
ADR4SRML: Synchronisation

- Prem: individual rules
- Lhs: piece of graph
ADR4SRML: Synchronisation

- Prem: individual rules
- Lhs: piece of graph
- Rhs: synchronised rhs’s
Outline

Overview

Advance 1: ADR4UML4SOA

Advance 2: ADR4SRML

Advance 3: Towards D5a.

Conclusion
D5a: The deliverable

Details

- Title: *Hierarchical Graph Models of Extended Core Calculi.*
- Due: month 36.
D5a: The deliverable

Details

- Title: *Hierarchical Graph Models of Extended Core Calculi*.
- Due: month 36.

Preliminary ideas

- Elaborate on ADR, Ranked Graphs, Bigraphs, SHR.
- $\pi$-calculus encoding in ADR+RankedGraphs mix.
- Sketching various solutions for CaSpiS.
D5.a: Sketch for $\pi$

\[ (\text{cast}) : M \to P \]

\[ \nu \_\_ : N \times P \to P \]

\[ _-\_ : P \times P \to P \]

\[ _+_- : M \times M \to M \]

\[ \text{nil} \]

\[ \tau \_ : A \times P \to M \]

\[ _!_\_ : N \times N \times P \to M \]

Rough ideas

\[ \Rightarrow \text{Structure ` a la ADR} \]

\[ \Rightarrow \text{Names ` a la RankedGraphs} \]

\[ \Rightarrow \text{No translation, just interpretation} \]

Similar sketches for CaSpiS.
D5.a: Sketch for $\pi$

Rough ideas

- Structure à la ADR

$\text{cast} : M \rightarrow P$

$\nu \cdot \cdot : N \times P \rightarrow P$

$\cdot \cdot \cdot : P \times P \rightarrow P$

$\cdot \cdot \cdot : M \times M \rightarrow M$

$\text{nil} : M$

$\tau \cdot \cdot : A \times P \rightarrow M$

$\cdot \cdot \cdot : N \times N \times P \rightarrow M$

$\cdot \cdot \cdot : P$

lla@unipi.it

Advances in Architectural Design Rewriting

lafuente@di.unipi.it
D5.a: Sketch for $\pi$

**Rough ideas**
- Structure à la ADR
- Names à la RankedGraphs
- No translation, just interpretation
D5.a: Sketch for $\pi$

**Rough ideas**

- Structure à la ADR
- Names à la RankedGraphs
- No translation, just interpretation

Similar sketches for CaSpiS.
Outline

Overview

Advance 1: ADR4UML4SOA

Advance 2: ADR4SRML

Advance 3: Towards D5a.

Conclusion
Concluding Remarks

We presented sketches of

- Semantics of UML4SOA Profile extension.
- Operational semantics of SRML specifications.
- Steps towards D5a.
Concluding Remarks

We presented sketches of

- Semantics of UML4SOA Profile extension.
- Operational semantics of SRML specifications.
- Steps towards D5a.

Next Step

- Formalise the presented sketches.
Pointers


