Architectural Design Rewriting

R. Bruni[1], A. Lluch Lafuente[1], U. Montanari[1], E. Tuosto[2]

[1]Department of Computer Science, University of Pisa
[2]Department of Computer Science, University of Leicester

14 June 2007

Outline

Premises and Promises

Premises

D5.3a Reconfigurations preserving architectural types and shapes.

Promises

"In future, results of Deliverable 5.3a will be applied to help in deducting rules for reconfiguration."

- D6.1a

"We will develop a formal model for [...] reconfiguration ..."

- Wiki's T5.3 description

"The objective of WP5 is to provide rigorous mathematical foundations for combining services, including [...] reconfiguration."

- Wiki's WP5 objective

Premises and Promises

Premises

D5.3a Reconfigurations preserving architectural types and shapes.

Promises

"*In future, results of Deliverable 5.3a will be applied to help in deducting rules for reconfiguration.*"

- D6.1a

"*We will develop a formal model for [. . .] reconfiguration . . .*"

- Wiki's T5.3 description

"*The objective of WP5 is to provide rigorous mathematical foundations for combining services, including [. . .] reconfiguration.*"

- Wiki's WP5 objective

The recipe

Ingredients

- 20mg (Dynamic) Software Architectures;
- 3 Architectural Styles;
- 10dl Reconfigurations;
- Some Graphs;
- QoS (at will).

Preparation

Put the ingredients together and mix.

What are (software) architectures?

The experts say...

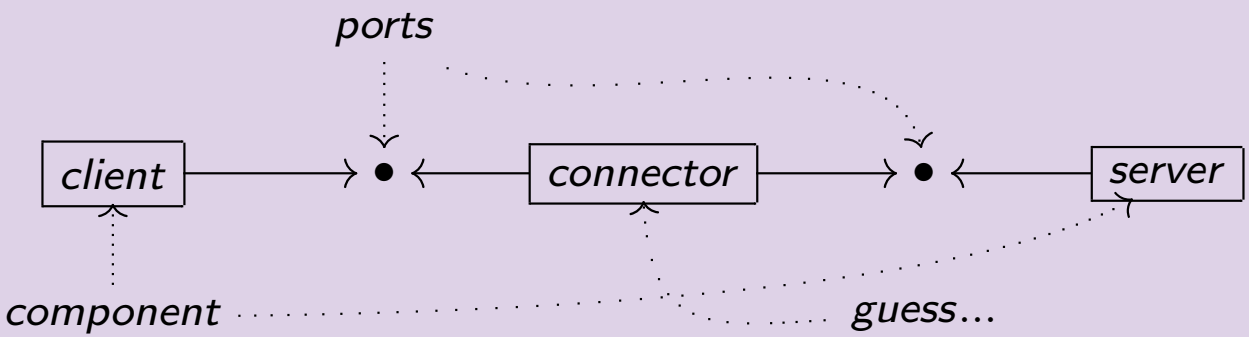
"... the structure of the components of a program / system, their interrelationship, and principles and guidelines governing their design and evolution over time."

- D. Garlan & D. Perry, 1995

A suitable model for architectures

Graphs as model of typical architectural models

- E.g. Components and Connectors.



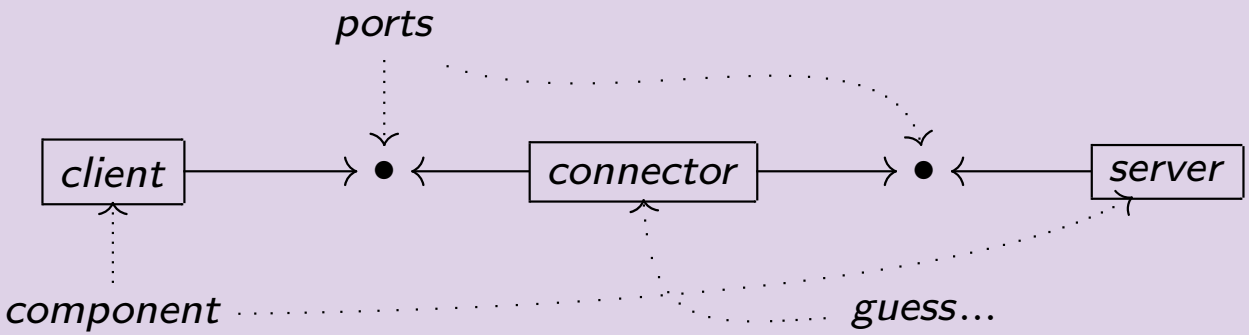
Graphs as model of Sensoria languages

- SRML diagrams (see SRML-P(v1.3)).
- Graphical encodings of process calculi.

A suitable model for architectures

Graphs as model of typical architectural models

- E.g. Components and Connectors.



Graphs as model of Sensoria languages

- SRML diagrams (see SRML-P(v1.3)).
- Graphical encodings of process calculi.

Architectural Styles

A style is . . .

". . . a set of patterns or rules for creating one or more architectures in a consistent fashion."

- IEEE standard 1471

Roughly... Style = Vocabulary + Rules

Benefits?

Understanding, Reuse, Construction, etc.

Architectural Styles

A style is . . .

". . . a set of patterns or rules for creating one or more architectures in a consistent fashion."

- IEEE standard 1471

Roughly... Style = Vocabulary + Rules

Benefits?

Understanding, Reuse, Construction, etc.

Models for styles

As a language

- Implicit style rules.
- Intuitive style-driven design.
- E.g. Graph grammars approaches [Le Métayer, 1996]

As additional constraints

- Explicit style rules.
- Complex structural rules easier to express.
- E.g. Alloy-based approaches [Garlan 2006].

Models for styles

As a language

- Implicit style rules.
- Intuitive style-driven design.
- E.g. Graph grammars approaches [Le Métayer, 1996]

As additional constraints

- Explicit style rules.
- Complex structural rules easier to express.
- E.g. Alloy-based approaches [Garlan 2006].

Reconfigurations

What?

A reconfiguration is a change in a (dynamic) architecture.

Why?

Security policies, load balancing, mobility, QoS assurance, etc.

E.g.?

Components joining/leaving the system, binding, wrapping, etc.

Reconfigurations

What?

A reconfiguration is a change in a (dynamic) architecture.

Why?

Security policies, load balancing, mobility, QoS assurance, etc.

E.g.?

Components joining/leaving the system, binding, wrapping, etc.

Reconfigurations

What?

A reconfiguration is a change in a (dynamic) architecture.

Why?

Security policies, load balancing, mobility, QoS assurance, etc.

E.g.?

Components joining/leaving the system, binding, wrapping, etc.

Outline

What is ADR?

$$\text{ADR} = \text{Term Rewriting} + \text{Designs}.$$
$$\text{Designs} = \text{Typed Graphs with Interfaces}.$$

Goal

A unifying model for . . .

- Design;
- Execution;
- Reconfiguration.

What is ADR?

$$\text{ADR} = \text{Term Rewriting} + \text{Designs}.$$
$$\text{Designs} = \text{Typed Graphs with Interfaces}.$$

Goal

A unifying model for ...

- Design;
- Execution;
- Reconfiguration.

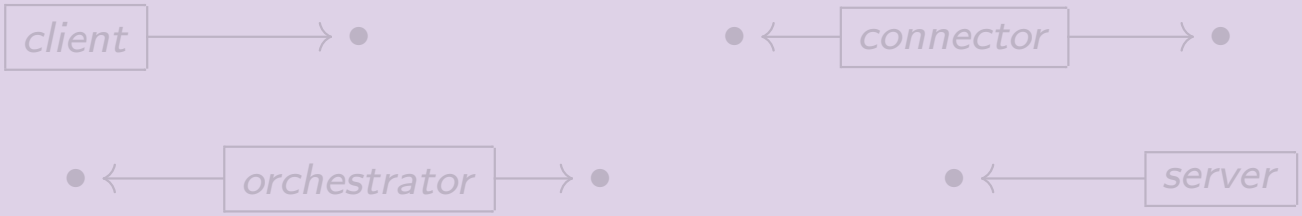
A Client-Server Example. Sorts

Sorts (Vocabulary of Architectural Elements)

The sorts of the client server style are

- Ports (nodes): $\bullet$.
- Component types (edges, designs)

- Actual components (edges)



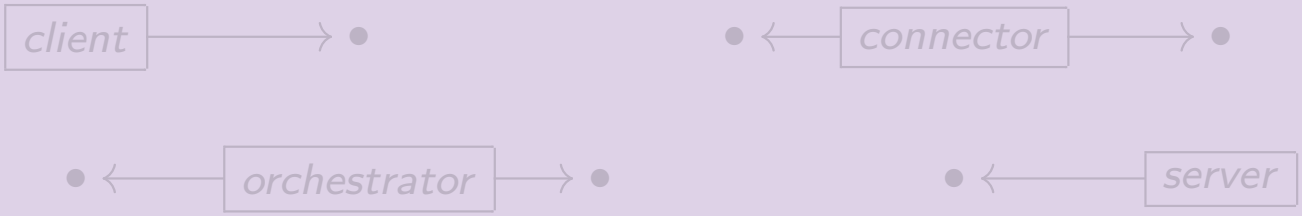
A Client-Server Example. Sorts

Sorts (Vocabulary of Architectural Elements)

The sorts of the client server style are

- Ports (nodes): $\bullet$.
- Component types (edges, designs)

- Actual components (edges)



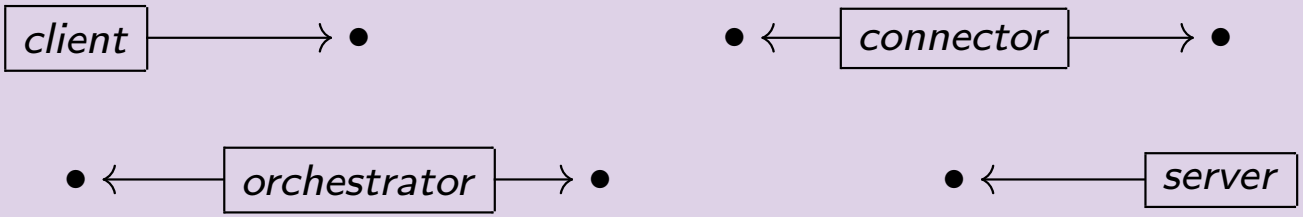
A Client-Server Example. Sorts

Sorts (Vocabulary of Architectural Elements)

The sorts of the client server style are

- Ports (nodes): $\bullet$.
- Component types (edges, designs)

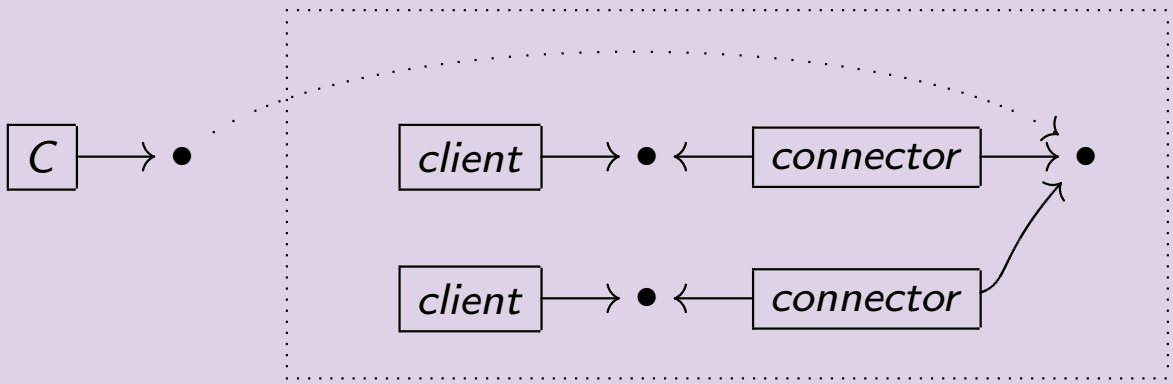
- Actual components (edges)



A Client-Server Example. Values

Values (Designs)

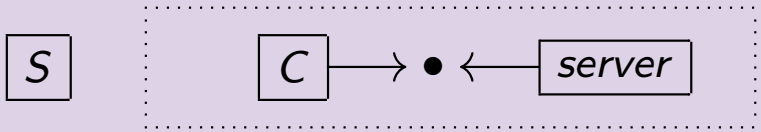
A design is a typed graph with an edge as interface.



... is a design of type C.

A Client-Server Example. Operations

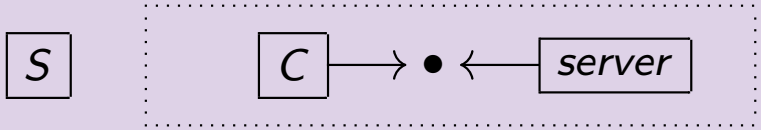
system : $C \longrightarrow S$



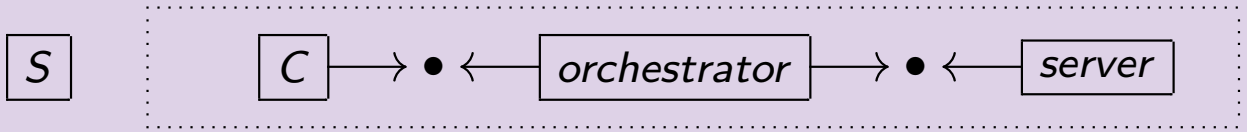
osystem : $C \longrightarrow S$

A Client-Server Example. Operations

system : $C \longrightarrow S$

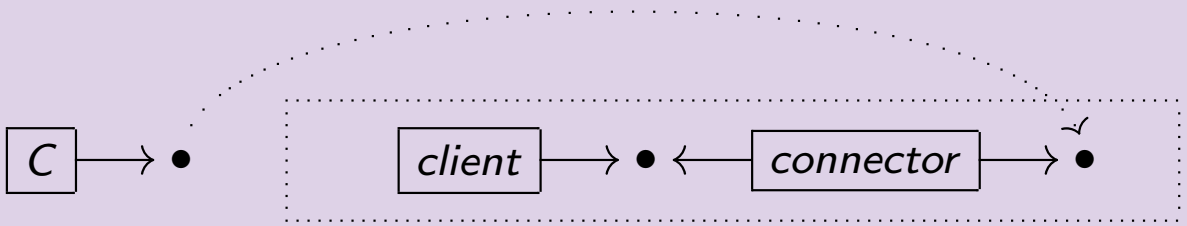


osystem : $C \longrightarrow S$



A Client-Server Example. Operations

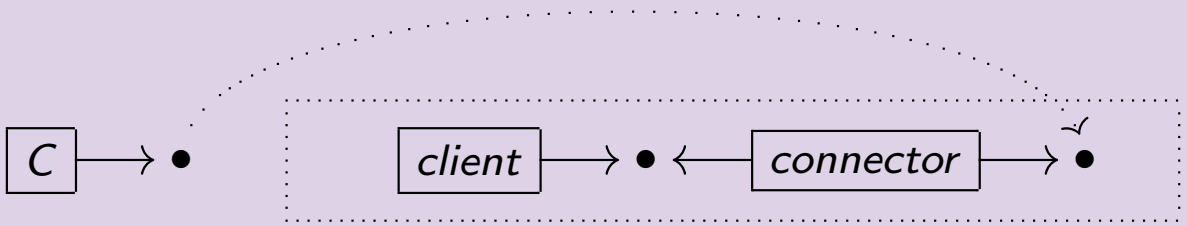
cclient $:\longrightarrow C$



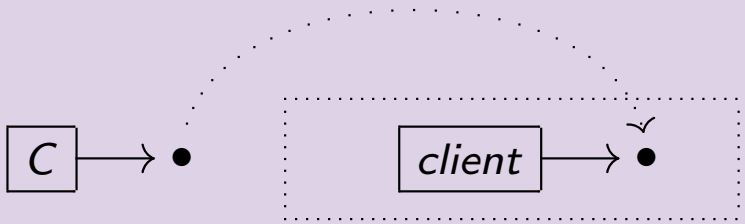
client $:\longrightarrow C$

A Client-Server Example. Operations

cclient :⟶ C

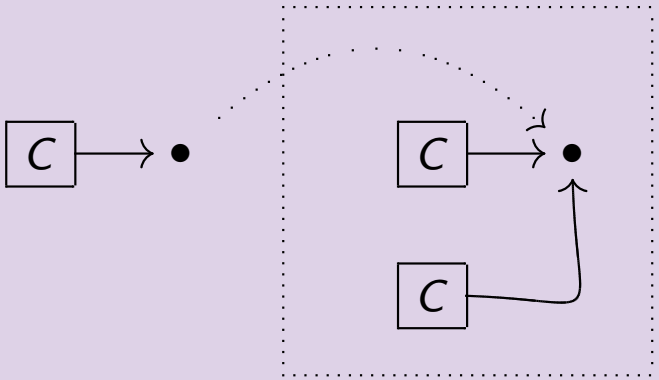


client :⟶ C



A Client-Server Example. Operations

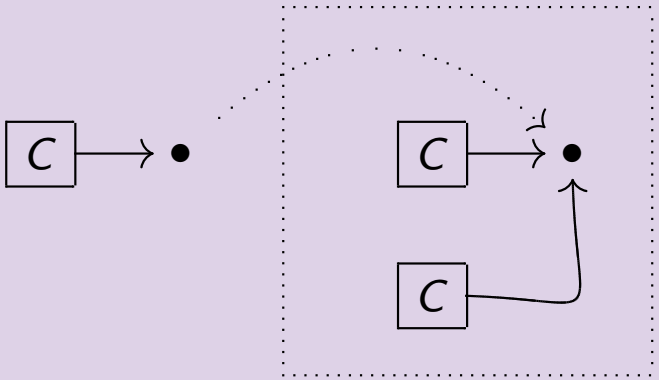
clients : $C \times C \longrightarrow C$



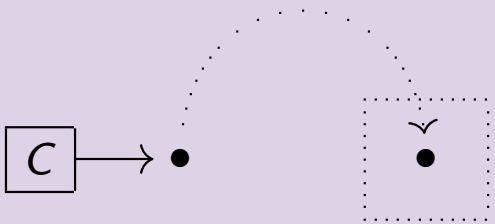
noclient : $\longrightarrow C$

A Client-Server Example. Operations

clients : $C \times C \longrightarrow C$



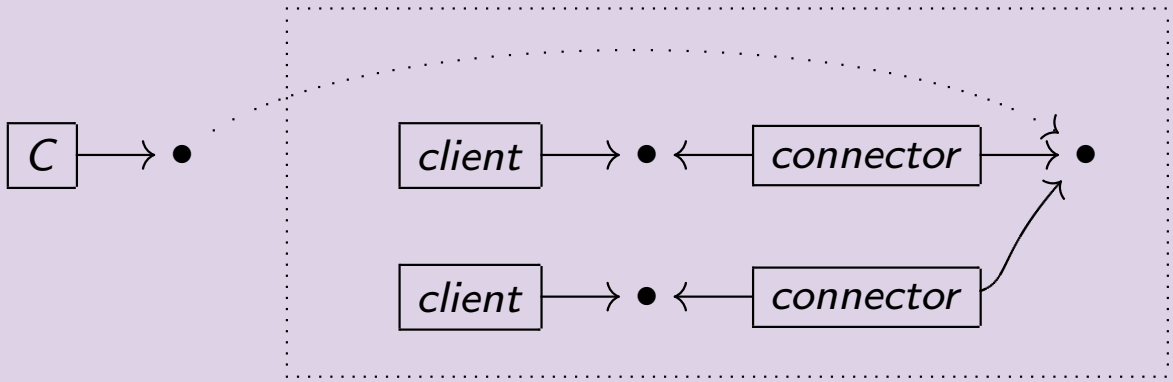
noclient : $\longrightarrow C$



A Client-Server Example. Values (again)

Values (evaluation of terms)

The value of `clients(cclient, cclient)` is ...



A Client-Server Example. Reconfiguration rules

join(Y)

$$X \longrightarrow \texttt{clients}(X, Y)$$

leave(Y)

$$\texttt{clients}(X, Y) \longrightarrow X$$

orchestrate

$$\texttt{system}(X) \longrightarrow \texttt{osystem}(X)$$

A Client-Server Example. Reconfiguration rules

join(Y)

$$X \longrightarrow \texttt{clients}(X, Y)$$

leave(Y)

$$\texttt{clients}(X, Y) \longrightarrow X$$

orchestrate

$$\texttt{system}(X) \longrightarrow \texttt{osystem}(X)$$

A Client-Server Example. Reconfiguration rules

join(Y)

$$X \longrightarrow \texttt{clients}(X, Y)$$

leave(Y)

$$\texttt{clients}(X, Y) \longrightarrow X$$

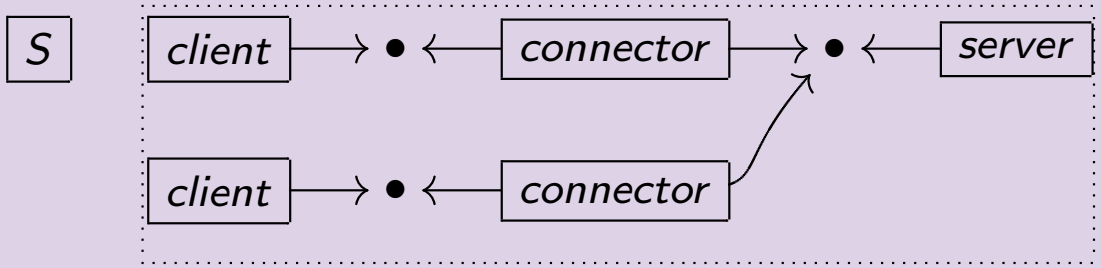
orchestrate

$$\texttt{system}(X) \longrightarrow \texttt{osystem}(X)$$

A Client-Server Example. A reconfiguration

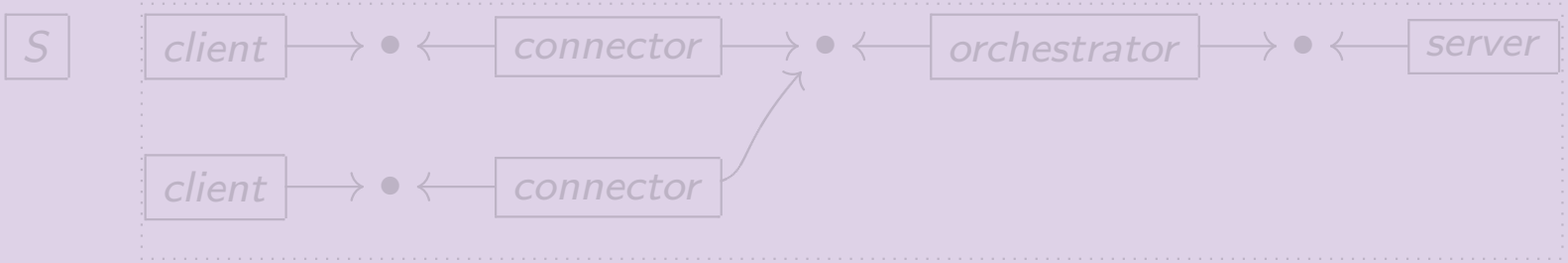
Applying rule orchestrate...

If we have system(clients(cclient, cclient)) ...



... X is matched with clients(cclient, cclient) ...

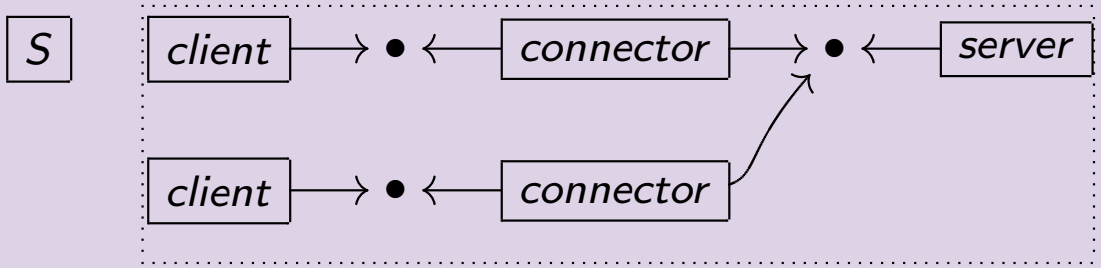
... and our system becomes osystem(clients(cclient, cclient)).



A Client-Server Example. A reconfiguration

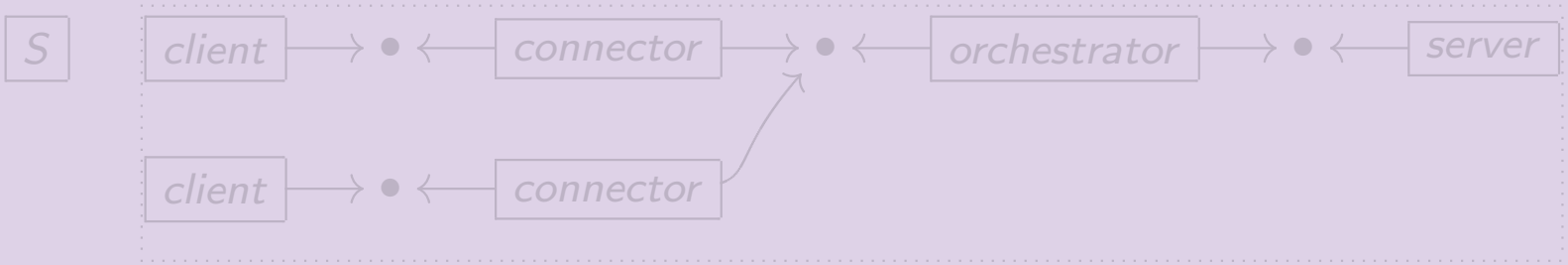
Applying rule orchestrate...

If we have system(clients(cclient, cclient)) ...



... X is matched with clients(cclient, cclient) ...

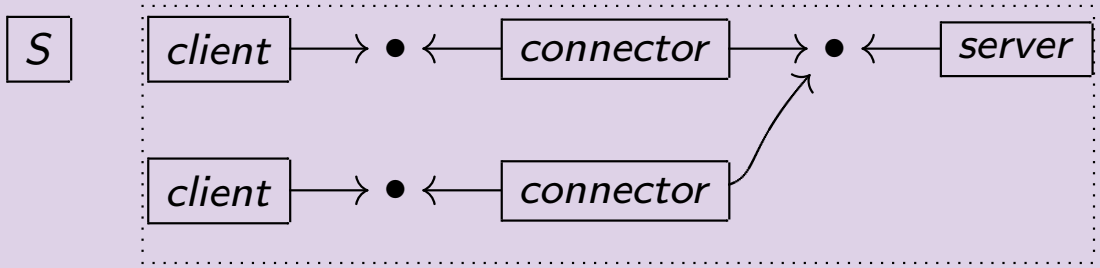
... and our system becomes osystem(clients(cclient, cclient)).



A Client-Server Example. A reconfiguration

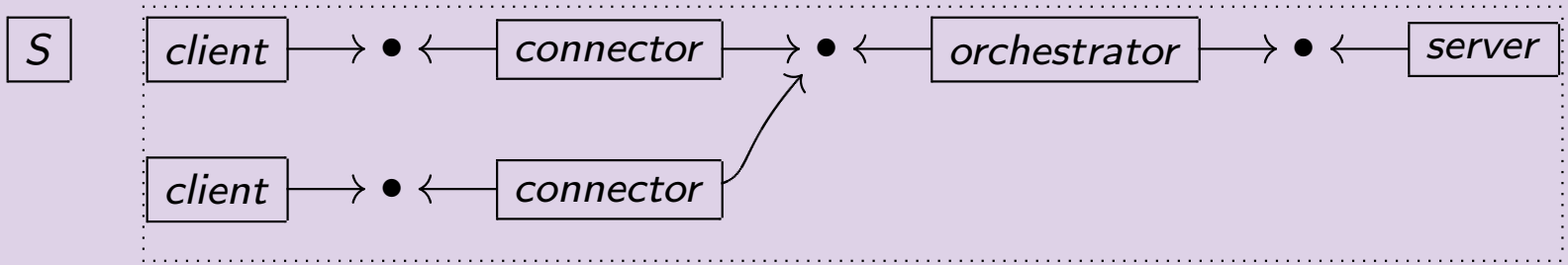
Applying rule `orchestrate`...

If we have $\texttt{system}(\texttt{clients}(\texttt{cclient}, \texttt{cclient}))$ ...



... X is matched with $\texttt{clients}(\texttt{cclient}, \texttt{cclient})$ ...

... and our system becomes $\texttt{osystem}(\texttt{clients}(\texttt{cclient}, \texttt{cclient}))$.



A Client-Server Example. More reconfigurations

disconnect

Labeled rule to model individual disconnection:

$$\texttt{cclient} \xrightarrow{\texttt{disc}} \texttt{client}$$

Dummy labeled rule:

$$\texttt{client} \xrightarrow{\texttt{disc}} \texttt{client}$$

Conditional labeled rule for complex disconnections:

$$\frac{X \xrightarrow{\texttt{disc}} X' \quad Y \xrightarrow{\texttt{disc}} Y'}{\texttt{clients}(X, Y) \xrightarrow{\texttt{disc}} \texttt{clients}(X', Y')}$$

A Client-Server Example. More reconfigurations

disconnect

Labeled rule to model individual disconnection:

$$\texttt{cclient} \xrightarrow{\texttt{disc}} \texttt{client}$$

Dummy labeled rule:

$$\texttt{client} \xrightarrow{\texttt{disc}} \texttt{client}$$

Conditional labeled rule for complex disconnections:

$$\frac{X \xrightarrow{\texttt{disc}} X' \quad Y \xrightarrow{\texttt{disc}} Y'}{\texttt{clients}(X, Y) \xrightarrow{\texttt{disc}} \texttt{clients}(X', Y')}$$

A Client-Server Example. More reconfigurations

disconnect

Labeled rule to model individual disconnection:

$$\texttt{cclient} \xrightarrow{\texttt{disc}} \texttt{client}$$

Dummy labeled rule:

$$\texttt{client} \xrightarrow{\texttt{disc}} \texttt{client}$$

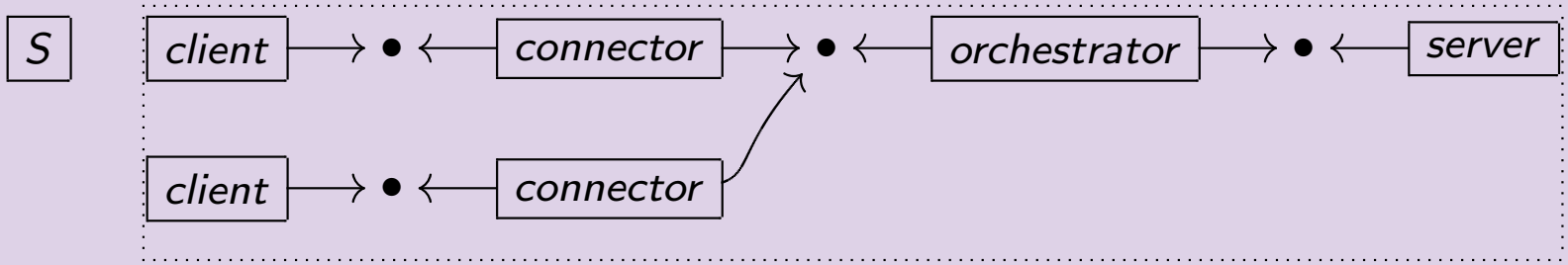
Conditional labeled rule for complex disconnections:

$$\frac{X \xrightarrow{\texttt{disc}} X' \quad Y \xrightarrow{\texttt{disc}} Y'}{\texttt{clients}(X, Y) \xrightarrow{\texttt{disc}} \texttt{clients}(X', Y')}$$

A Client-Server Example. Another reconfiguration

Applying rule `disconnect`...

If we have $\texttt{osystem}(\texttt{clients}(\texttt{cclient}, \texttt{cclient}))$ ...

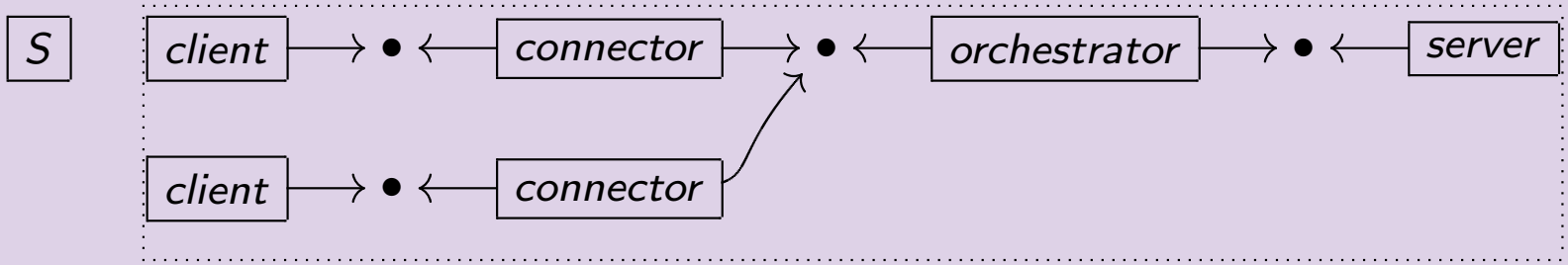


...we obtain $\texttt{osystem}(\texttt{clients}(\texttt{client}, \texttt{client}))$.

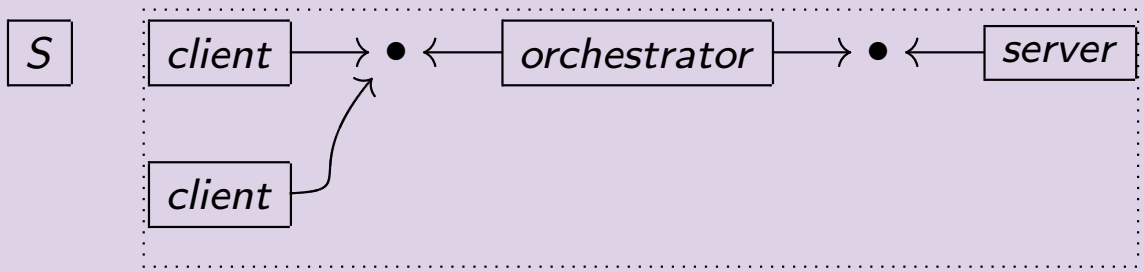
A Client-Server Example. Another reconfiguration

Applying rule `disconnect`...

If we have $\texttt{osystem}(\texttt{clients}(\texttt{cclient}, \texttt{cclient}))$ ...



... we obtain $\texttt{osystem}(\texttt{clients}(\texttt{client}, \texttt{client}))$.



Constrained Designs

Designs with Constraints

- Nodes as QoS attributes.
- Edges as (c-semiring based) constraints.

Purpose

- Postpone design decisions.
- Trigger reconfigurations.
- Measure reconfiguration alternatives.

Constrained Designs

Designs with Constraints

- Nodes as QoS attributes.
- Edges as (c-semiring based) constraints.

Purpose

- Postpone design decisions.
- Trigger reconfigurations.
- Measure reconfiguration alternatives.

ADR in detail

ADR is . . .

- Sorts: Vocabulary, Types (Edge and node labels).
- Values: Designs (graphs with interfaces).
- Operations: grammar-like style rules.
- Terms: proofs of construction.
- Terms (with variables): partial Designs.
- Axioms: properties of operations.
- Membership equations: additional style rules.
- Rewriting rules: behaviour, reconfigurations.
- Rewriting strategies: style conformance, style analysis, etc.

Outline

What next?

Ongoing...

- Implementation in Maude.
- Style for a graphical encoding of the π-calculus.
- Styles for SHR variants.

Future... (Sensorize the Approach)

- Not just symbolic types: service, behavioural, spatial, etc.
- Application to SRML architectures.
- Case Study / Scenario.

What next?

Ongoing...

- Implementation in Maude.
- Style for a graphical encoding of the π-calculus.
- Styles for SHR variants.

Future... (Sensorize the Approach)

- Not just symbolic types: service, behavioural, spatial, etc.
- Application to SRML architectures.
- Case Study / Scenario.

What next?

Ongoing...

- Implementation in Maude.
- Style for a graphical encoding of the π-calculus.
- Styles for SHR variants.

Future... (Sensorize the Approach)

- Not just symbolic types: service, behavioural, spatial, etc.
- Application to SRML architectures.
- Case Study / Scenario.

What next?

Ongoing...

- Implementation in Maude.
- Style for a graphical encoding of the π-calculus.
- Styles for SHR variants.

Future... (Sensorize the Approach)

- Not just symbolic types: service, behavioural, spatial, etc.
- Application to SRML architectures.
- Case Study / Scenario.

What next?

Ongoing...

- Implementation in Maude.
- Style for a graphical encoding of the π-calculus.
- Styles for SHR variants.

Future... (Sensorize the Approach)

- Not just symbolic types: service, behavioural, spatial, etc.
- Application to SRML architectures.
- Case Study / Scenario.

What next?

Ongoing...

- Implementation in Maude.
- Style for a graphical encoding of the π-calculus.
- Styles for SHR variants.

Future... (Sensorize the Approach)

- Not just symbolic types: service, behavioural, spatial, etc.
- Application to SRML architectures.
- Case Study / Scenario.

What next?

Ongoing...

- Implementation in Maude.
- Style for a graphical encoding of the π-calculus.
- Styles for SHR variants.

Future... (Sensorize the Approach)

- Not just symbolic types: service, behavioural, spatial, etc.
- Application to SRML architectures.
- Case Study / Scenario.

Further reading

Two drafts

- Style-Based Reconfigurations of Software Architectures with QoS Constraints.
- Architectural Styles for Graphical Encodings of Process Algebras.

I got the idea but I have some...

- Questions?
- Remarks?
- Criticism?
- Suggestions?