Intentional Automata

Modelling $\mathcal{R}$eo connectors

David Costa
costa@cwi.nl

CWI

Amsterdam, October 23 - 2007

Outline

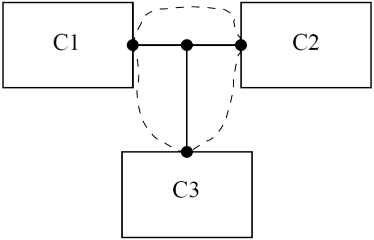
1. Motivation
   - Connectors for Composite Systems/Services
   - Previous Work
2. Intentional Automata
   - Concepts and Definition
3. Modelling Reo
   - Reo class of Intentional Automata - RIA
   - Examples: Primitive Connector
   - Reo Join and Hide operations
   - Example: Composite Connector

Outline

Composite Systems/Services

Our interest in Connectors

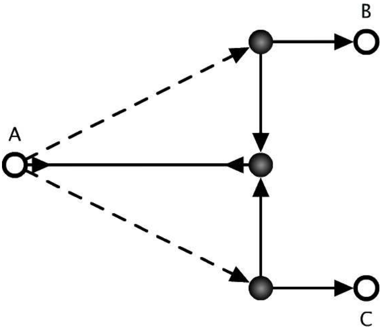
models, methods, tools and techniques;



Composite Systems/Services

Our interest in Connectors

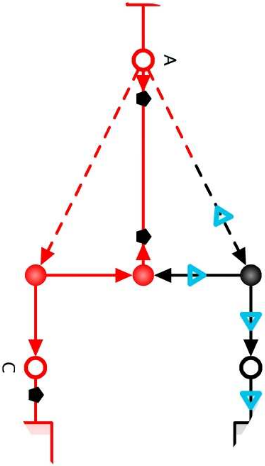
Connector design and specification



Composite Systems/Services

Our interest in Connectors

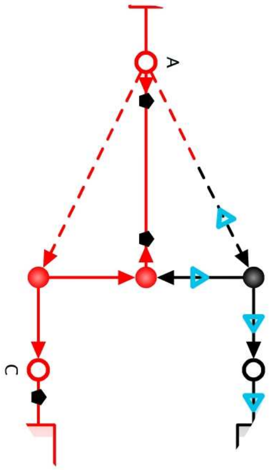
Connector Synthesis/Code generation



Composite Systems/Services

Our interest in Connectors

Formal analysis of non-functional properties



Outline

Automata formalisms

Interface Automata [Alfaro, Hezinger '01]

- Automata model that captures *temporal* aspects of software components interface.
- Input assumptions about the order in which the methods of a component are called;
- Output assumptions about the order in which the component calls external methods;
- Automatic compatibility check;
- Can be seen as a type system for component interaction.

Automata formalisms

Constraint Automata [BSAR'04]

We have heard a lot about it on yesterday's afternoon session.

- automata model used to specify the operational behaviour of $\mathcal{R}eo$ connectors;
- synchronization constraints are encoded by transitions that fire when satisfying a given data constraint;
- automata operators product and hide model $\mathcal{R}eo$ operators join and hiding.
- Well know techniques of finite automata for equivalence and simulation are adapted to Constraint Automata.

Automata formalisms

CA falls short when it comes:

- to describe context-sensitive connectors;
- in providing an accurate level of abstraction to reason about non-functional properties.

Connector Colouring [CCA'05]

- Provides accurate semantics for context-sensitive $\mathcal{R}eo$ connectors;
- And models $\mathcal{R}eo$ join operation to allow composition of context-sensitive connectors.

Outline

Our view of a connector

Connector

1. is a black box;
2. with an interface defined by a set of ports $\Sigma \subseteq \mathcal{N}ames = \{A, B, C, \ldots\}$;
3. the connector degree is given by $\sharp\Sigma$;
   - $\mathcal{R}eo$ channels are connectors of degree 2;
   - merger and replicator are connectors of degree 3.

Port

1. A port is an interaction point;
2. We associate a name $N \in \mathcal{N}ames$ to a port.

Our view of a connector

Connector

1. is a black box;
2. with an interface defined by a set of ports $\Sigma \subseteq \mathcal{N}ames = \{A, B, C, \ldots\}$;
3. the connector degree is given by $\sharp\Sigma$;
   - $\mathcal{R}eo$ channels are connectors of degree 2;
   - merger and replicator are connectors of degree 3.

Port

1. A port is an interaction point;
2. We associate a name $N \in \mathcal{N}ames$ to a port.

Our view of a connector

Connector

1. is a black box;
2. with an interface defined by a set of ports $\Sigma \subseteq \mathcal{N}ames = \{A, B, C, \ldots\}$;
3. the connector degree is given by $\sharp\Sigma$;
   - $\mathcal{R}eo$ channels are connectors of degree 2;
   - merger and replicator are connectors of degree 3.

Port

1. A port is an interaction point;
2. We associate a name $N \in \mathcal{N}ames$ to a port.

Our view of a connector

Connector

1. is a black box;
2. with an interface defined by a set of ports $\Sigma \subseteq \mathcal{N}ames = \{A, B, C, \ldots\}$;
3. the connector degree is given by $\sharp\Sigma$;
   - $\mathcal{R}eo$ channels are connectors of degree 2;
   - merger and replicator are connectors of degree 3.

Port

1. A port is an interaction point;
2. We associate a name $N \in \mathcal{N}ames$ to a port.

Request and Firing events

Request

- The environment can interact with a port of a connector performing on it a request.

Request-set or Experiment

- Given a connector C with a set of ports Σ;
- A request-set or experiment is a subset $R \subseteq \Sigma$ of the set of ports;
- The different ways the environment can interact with a connector is given by the set of all request-sets $\mathscr{R} = \mathcal{P}\Sigma$;
- We call $\mathscr{R}$ the experiments of C.

Request and Firing events

Request

- The environment can interact with a port of a connector performing on it a request.

Request-set or Experiment

- Given a connector C with a set of ports Σ;
- A request-set or experiment is a subset $R \subseteq \Sigma$ of the set of ports;
- The different ways the environment can interact with a connector is given by the set of all request-sets $\mathscr{R} = \mathcal{P}\Sigma$;
- We call $\mathscr{R}$ the experiments of C.

Example

Synchronous channel

Sync – A synchronous channel with the set of ports $\Sigma = \{A, B\}$

- The experiments of *Sync* are given by the set $\mathcal{P}\Sigma$;
- $\mathscr{R} = \{\emptyset, \{A\}, \{B\}, \{A, B\}\}$

Example

Synchronous channel

Sync – A synchronous channel with the set of ports $\Sigma = \{A, B\}$

- The experiments of *Sync* are given by the set $\mathcal{P}\Sigma$;
- $\mathscr{R} = \{\emptyset, \{A\}, \{B\}, \{A, B\}\}$

empty request-set, $\emptyset$

The empty request-set $\emptyset$ denotes the empty experiment–none of the ports of *Sync* receives a request from the environment;

Example
Synchronous channel

Sync – A synchronous channel with the set of ports $\Sigma = \{A, B\}$

- The experiments of *Sync* are given by the set $\mathcal{P}\Sigma$;
- $\mathscr{R} = \{\emptyset, \{A\}, \{B\}, \{A, B\}\}$

request-set $\{A\}$

The request-set $\{A\}$ denotes the experiment that involves receiving a request on port A and not on port B;

Example

Synchronous channel

Sync – A synchronous channel with the set of ports $\Sigma = \{A, B\}$

- The experiments of *Sync* are given by the set $\mathcal{P}\Sigma$;
- $\mathscr{R} = \{\emptyset, \{A\}, \{B\}, \{A, B\}\}$

request-set $\{B\}$

Similarly, the request-set $\{B\}$ denotes the experiment that involves receiving a request on port B but not on port A;

Example

Synchronous channel

Sync – A synchronous channel with the set of ports $\Sigma = \{A, B\}$

- The experiments of *Sync* are given by the set $\mathcal{P}\Sigma$;
- $\mathscr{R} = \{\emptyset, \{A\}, \{B\}, \{A, B\}\}$

request-set $\{A, B\}$

Finally the request-set $\{A, B\}$ denotes an experiment that involves receiving requests on both ports A and B simultaneously.

Request and Firing events

Firing

- A connector can fire a port allowing data to flow through the port;

Firing-set

- A firing-set is a subset $F \subseteq \Sigma$ of the set of ports;
- The set of all possible firing-set of a connector is denoted by $\mathscr{F}$ ($\mathscr{F} \subseteq \mathcal{P}\Sigma$);
- We call $\mathscr{F}$ the firings of the connector.

Back to our example

Synchronous channel

Sync – A synchronous channel with the set of ports $\Sigma = \{A, B\}$

- The firings of *Sync* are given by the set:
- $\mathscr{F} = \{\emptyset, \{A, B\}\}$

empty firing-set, $\emptyset$

The empty firing-set $\emptyset$ denotes *quiescence*–no firing at any of the ports of *Sync*;

Back to our example

Synchronous channel

Sync – A synchronous channel with the set of ports $\Sigma = \{A, B\}$

- The firings of *Sync* are given by the set:
- $\mathscr{F} = \{\emptyset, \{A, B\}\}$

request-set $\{A, B\}$

The firing-set $\{A, B\}$ denotes the simultaneous firing of ports A and B. Implying simultaneous data-flow in the two ports.

Interaction Step

Transition

request-set | firing-set

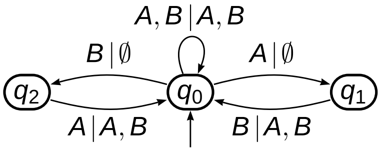
- A connector processes one (possibly empty) request-set $R \in \mathscr{R}$ and produces a (possibly empty) firing-set $F \in \mathscr{F}$ in each interaction step;

Internal transition

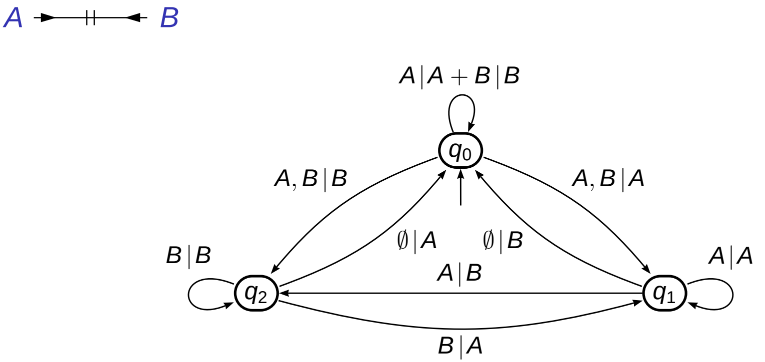
- A interaction step $\emptyset \mid \emptyset$ that involves the empty request-set and the empty firing-set, involves none of the ports of the automaton and is thus an internal transition.

Example

$A \longrightarrow B$



Non-deterministic example



Formal definition

Non-deterministic Intentional Automata

Definition

A non-deterministic intentional automaton over the set of ports Σ is a system $\mathcal{A} = (Q, \mathscr{R}, \mathscr{F}, \delta)$, with

- a set of states Q;
- a transition function $\delta : Q \to \mathcal{P}(\mathscr{F} \times Q)^{\mathscr{R}}$ that associates for every state $q \in Q$,
  - a function $\delta(q) \in \mathcal{P}(\mathscr{F} \times Q)^{\mathscr{R}}$, that is, $\delta(q) : \mathscr{R} \longrightarrow \mathcal{P}(\mathscr{F} \times Q)$ where
  - $\mathscr{R}$ and $\mathscr{F}$ are respectively the experiments and firings of $\mathcal{A}$.

Formal definition

Non-deterministic Intentional Automata

Optionally

- $\mathcal{A}$ can have a distinguished set of initial states $I \subseteq Q$;
- in this case the intentional automaton is denoted by $\mathcal{A} = (Q, \mathscr{R}, \mathscr{F}, \delta, I)$.
- If no set of initial states is defined then all states in Q are initial.

Equivalence of automata and Operations on automata

Notions of equivalence

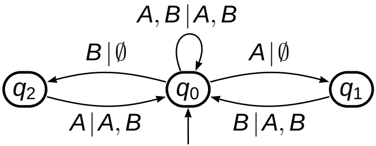
CCS-like notion of strong and weak equivalence

Operations on automata

- Restriction (enables to limit the number of connectors or components that can connect to a port);
- Hiding (conceal the internal structure of the composite connector, namely internal ports and internal interactions);
- Product (combination of interleaving and synchronous composition);

What makes an Intentional Automata a valid model in Reo?

Intentional automata model for *Sync*

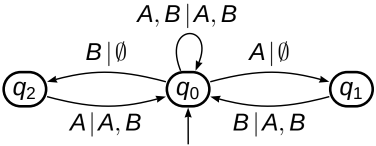


We start by considering structured states in the automata

(s,P) where s refers to the actual state of the connector and P is the set of pending ports.

What makes an Intentional Automata a valid model in Reo?

Intentional automata model for *Sync*

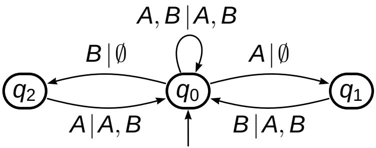


From abstract states to structured states

- $q_0 \xrightarrow{A|\emptyset} q_1$
- $q_1 \xrightarrow{B|A,B} q_0$

What makes an Intentional Automata a valid model in Reo?

Intentional automata model for *Sync*

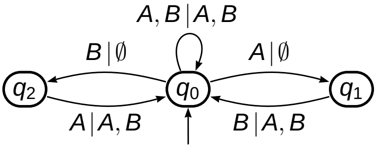


From abstract states to structured states

- $(s, \emptyset) \xrightarrow{A|\emptyset} q_1$
- $q_1 \xrightarrow{B|A,B} (s, \emptyset)$

What makes an Intentional Automata a valid model in Reo?

Intentional automata model for *Sync*

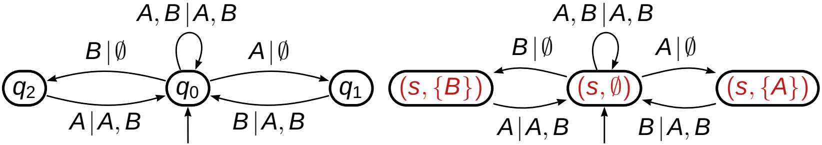


From abstract states to structured states

- $(s, \emptyset) \xrightarrow{A|\emptyset} (s, \{A\})$
- $(s, \{A\}) \xrightarrow{B|A,B} (s, \emptyset)$

What makes an Intentional Automata a valid model in Reo?

Intentional automata model for *Sync*



From abstract states to structured states

- $(s, \emptyset) \xrightarrow{A|\emptyset} (s, \{A\})$
- $(s, \{A\}) \xrightarrow{B|A,B} (s, \emptyset)$

Outline

1. Motivation
   - Connectors for Composite Systems/Services
   - Previous Work
2. Intentional Automata
   - Concepts and Definition
3. Modelling Reo
   - Reo class of Intentional Automata - RIA
   - Examples: Primitive Connector
   - Reo Join and Hide operations
   - Example: Composite Connector

Reo class of Intentional Automata

Definition

A $\mathcal{R}eo$ intentional automaton over the set of ports Σ, and the set of connector states S is a non-deterministic intentional automaton $A = (Q, \mathscr{R}, \mathscr{F}, \delta)$, with

- the set of configurations $Q = S \times \mathcal{P}\Sigma$ and
- the transition function $\delta : Q \to \mathcal{P}(\mathscr{F} \times Q)^{\mathscr{R}}$ that associates with every state $q = (s, P) \in Q$ a function $\delta_q : \mathscr{R}|_P \longrightarrow \mathcal{P}_{ne}(\mathscr{F} \times Q)$ such that

$$\boxed{\delta_{(s,P)}(R) = \delta_{(s,\emptyset)}(R \cup P).}$$

Configuration Table

Semantic property

$$\delta_{(s,P)}(R) = \delta_{(s,\emptyset)}(R \cup P)$$

Outline

Lossysync

intentional automaton $Lossysync = (Q, \mathscr{R}, \mathscr{F}, \delta, I)$

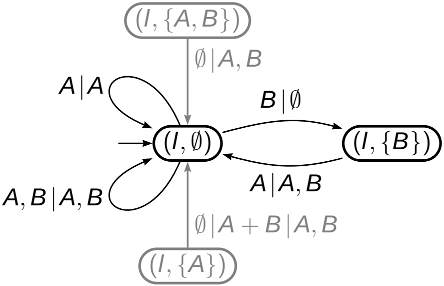
- $S = \{I\}$,
- $\mathscr{R} = \mathcal{P}\{A, B\}$,
- $\mathscr{F} = \{\emptyset, \{A\}, \{A, B\}\}$,
- and δ is given by the configuration table I:

$s \backslash R$	$\emptyset$	$\{A\}$	$\{B\}$	$\{A, B\}$
I	$\langle \emptyset, (I, \emptyset) \rangle$	$\langle \{A\}, (I, \emptyset) \rangle$	$\langle \emptyset, (I, \{B\}) \rangle$	$\langle \{A, B\}, (I, \emptyset) \rangle$

- $I = \{(I, \emptyset)\}$

Lossysync

The respective labelled transition diagram



$FIFO_1$

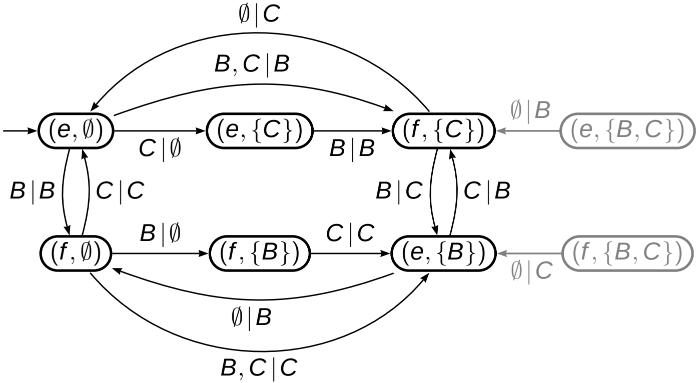
intentional automaton $FIFO_1 = (Q, \mathscr{R}, \mathscr{F}, \delta, I)$

- $S = \{e, f\}$,
- $\mathscr{R} = \mathcal{P}\{B, C\}$,
- $\mathscr{F} = \{\emptyset, \{B\}, \{C\}\}$,
- and δ is given by the following configuration table:

S \ R	$\emptyset$	$\{B\}$	$\{C\}$	$\{B, C\}$
e	$\langle\emptyset, (e, \emptyset)\rangle$	$\langle\{B\}, (f, \emptyset)\rangle$	$\langle\emptyset, (e, \{C\})\rangle$	$\langle\{B\}, (f, \{C\})\rangle$
f	$\langle\emptyset, (f, \emptyset)\rangle$	$\langle\emptyset, (f, \{B\})\rangle$	$\langle\{C\}, (e, \emptyset)\rangle$	$\langle\{C\}, (e, \{B\})\rangle$

- $I = \{(e, \emptyset)\}$

$FIFO_1$


$\emptyset \mid C$
$B, C \mid B$
$(e, \emptyset)$
$(e, \{C\})$
$(f, \{C\})$
$\emptyset \mid B$
$(e, \{B, C\})$
$C \mid \emptyset$
$B \mid B$
$B \mid B$
$C \mid C$
$B \mid C$
$C \mid B$
$B \mid \emptyset$
$C \mid C$
$(f, \emptyset)$
$(f, \{B\})$
$(e, \{B\})$
$(f, \{B, C\})$
$\emptyset \mid C$
$\emptyset \mid B$
$B, C \mid C$

Outline

1. Motivation
   - Connectors for Composite Systems/Services
   - Previous Work
2. Intentional Automata
   - Concepts and Definition
3. Modelling Reo
   - Reo class of Intentional Automata - RIA
   - Examples: Primitive Connector
   - Reo Join and Hide operations
   - Example: Composite Connector

Reo *join* and *hide* operations

Semantics

$$[\![\ \mathit{Join}\ ]\!] = \mathit{restrict}\ \cdot\ \mathit{product}$$
$$[\![\ \mathit{Hide}\ ]\!] = \mathit{hiding}$$

Outline

1. Motivation
   - Connectors for Composite Systems/Services
   - Previous Work
2. Intentional Automata
   - Concepts and Definition
3. Modelling Reo
   - Reo class of Intentional Automata - RIA
   - Examples: Primitive Connector
   - Reo Join and Hide operations
   - Example: Composite Connector

$(Lossysync \bowtie FIFO_1)[B]$

- $S = \{le, lf\}$,
- $\mathscr{R} = \mathcal{P}\{A, C\}$,
- $\mathscr{F} = \{\emptyset, \{A\}, \{C\}\}$,
- and δ is given by the following configuration tables e and f:

s \ R	$\emptyset$	$\{A\}$	$\{C\}$	$\{A, C\}$
le	$\langle\emptyset, (le, \emptyset)\rangle$	$\langle\{A\}, (lf, \emptyset)\rangle$	$\langle\emptyset, (le, \{C\})\rangle$	$\langle\{A\}, (lf, \{C\})\rangle$
lf	$\langle\emptyset, (lf, \emptyset)\rangle$	$\langle\{A\}, (lf, \emptyset)\rangle$	$\langle\{C\}, (le, \emptyset)\rangle$	$\langle\{A, C\}, (le, \emptyset)\rangle$

- $I = \{(le, \emptyset)\}$