Choreography Automata

Franco Barbanera[1], Ivan Lanese[2], Emilio Tuosto[3]

[1] University of Catania
[2] University of Bologna/INRIA
[3] GSSI/University of Leicester

Coordination@DisCoTec 2020 - Malta, June 2020

Good ideas are recyclable

If you have a bunch of dancers...

Good ideas are recyclable

If you have a bunch of dancers...

....would you like to end up with this....

or with THIS?

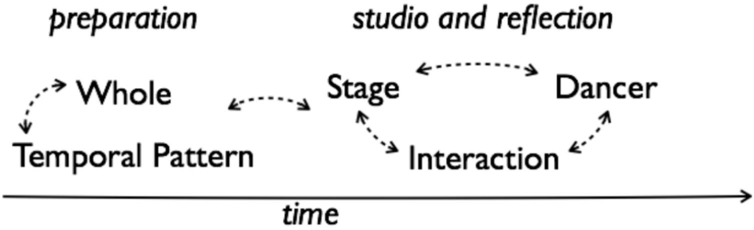
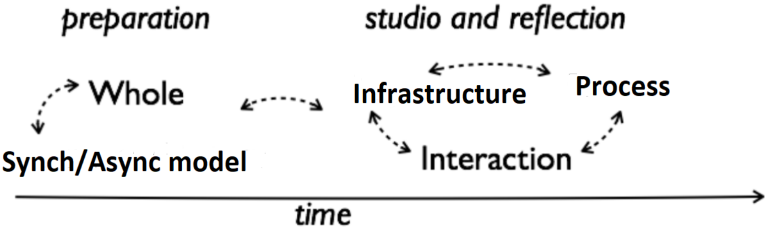
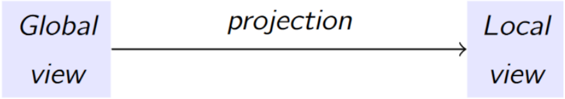


Figure 10. Focal points along the creative phases.

The recycling



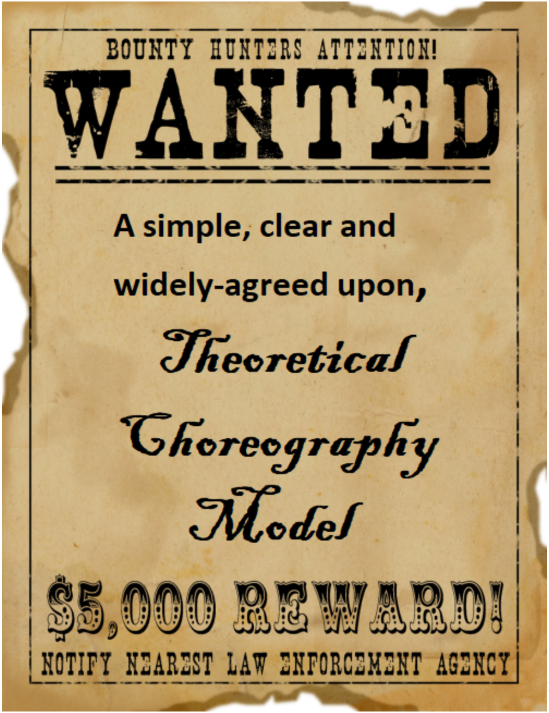
More abstractly:
coexistence of two distinct but related views of a system: the *global* and the *local* views.



projection is an operation producing the local view from the global one

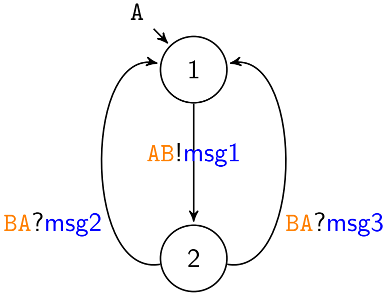
The choreographic approach: A lighthouse on the Formal Verification road

- specification languages: WS-CDL, BPMN, ...
- choreographies for microservices;
- experimental choreographic langauges: Chor
- etc.


BOUNTY HUNTERS ATTENTION!
WANTED
A simple, clear and
widely-agreed upon,
Theoretical
Choreography
Model
$5,000 REWARD!
NOTIFY NEAREST LAW ENFORCEMENT AGENCY

Which abstraction for processes?

Which abstraction for processes?



Communicating Finite State Machines (CFSMs)

A formalism for the description and the analysis of distributed systems.

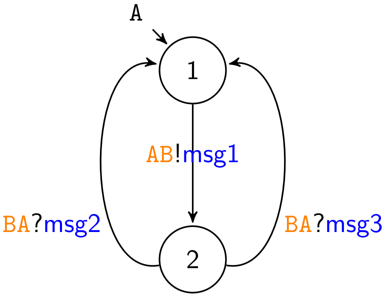
A machine $M_{\mathtt{A}}$

- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
  asynchronously; *through the directed buffered FIFO channel* AB
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
  through channel BA;
- and so on....

Communicating Finite State Machines (CFSMs)

A formalism for the description and the analysis of distributed systems.

A machine $M_{\mathtt{A}}$

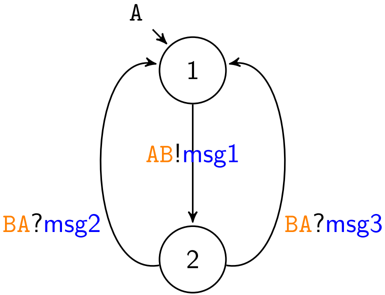


- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
  asynchronously; *through the directed buffered FIFO channel* AB
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
  through channel BA;
- and so on....

Communicating Finite State Machines (CFSMs)

A formalism for the description and the analysis of distributed systems.

A machine $M_{\mathtt{A}}$

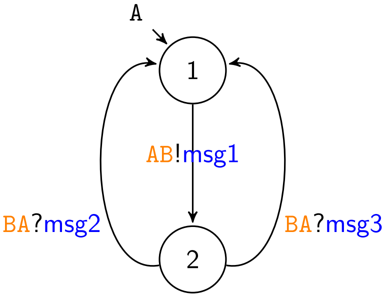


- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
  asynchronously; *through the directed buffered FIFO channel* AB
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
  through channel BA;
- and so on....

Communicating Finite State Machines (CFSMs)

A formalism for the description and the analysis of distributed systems.

A machine $M_{\mathtt{A}}$

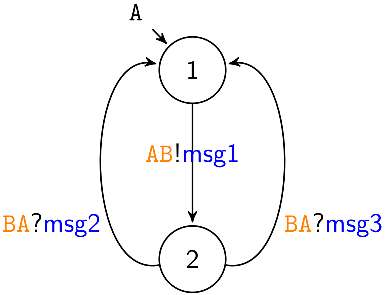


- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
  asynchronously; *through the directed buffered FIFO channel* AB
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
  through channel BA;
- and so on....

Communicating Finite State Machines (CFSMs)

A formalism for the description and the analysis of distributed systems.

A machine $M_{\mathtt{A}}$



- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
  asynchronously; *through the directed buffered FIFO channel* AB
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
  through channel BA;
- and so on....

Systems of CFSMs

A *system* of CFSMs:

$$S = (M_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$$

- $\mathbf{P}$ is the set of *roles* (participants) of S, and
- for each $\mathrm{p} \in \mathbf{P}$, $M_{\mathrm{p}} = (Q_{\mathrm{p}}, q_{0\mathrm{p}}, \mathbb{A}, \delta_{\mathrm{p}})$ is a CFSM.

A *configuration* of S:

$$s = (\vec{q}, \vec{w})$$

- $\vec{q} = (q_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$ *the overall state of the system*
  where $q_{\mathrm{p}} \in Q_{\mathrm{p}}$ *the current state of machine* M_{p}
- $\vec{w} = (w_{\mathrm{pq}})_{\mathrm{pq} \in \mathsf{Chan}}$ with $w_{\mathrm{pq}} \in \mathbb{A}^*$. *the current contents of channels*

The initial configuration of S is $s_0 = (\vec{q_0}, \vec{\varepsilon})$ with $\vec{q_0} = (q_{0_{\mathrm{p}}})_{\mathrm{p} \in \mathbf{P}}$.

Systems of CFSMs

A *system* of CFSMs:

$$S = (M_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$$

- $\mathbf{P}$ is the set of *roles* (participants) of S, and
- for each $\mathrm{p} \in \mathbf{P}$, $M_{\mathrm{p}} = (Q_{\mathrm{p}}, q_{0\mathrm{p}}, \mathbb{A}, \delta_{\mathrm{p}})$ is a CFSM.

A *configuration* of S:

$$s = (\vec{q}, \vec{w})$$

- $\vec{q} = (q_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$ *the overall state of the system*
  where $q_{\mathrm{p}} \in Q_{\mathrm{p}}$ *the current state of machine* M_{p}
- $\vec{w} = (w_{\mathrm{pq}})_{\mathrm{pq} \in \mathsf{Chan}}$ with $w_{\mathrm{pq}} \in \mathbb{A}^*$. *the current contents of channels*

The initial configuration of S is $s_0 = (\vec{q_0}, \vec{\varepsilon})$ with $\vec{q_0} = (q_{0_{\mathrm{p}}})_{\mathrm{p} \in \mathbf{P}}$.

Systems of CFSMs

A *system* of CFSMs:

$$S = (M_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$$

- $\mathbf{P}$ is the set of *roles* (participants) of S, and
- for each $\mathrm{p} \in \mathbf{P}$, $M_{\mathrm{p}} = (Q_{\mathrm{p}}, q_{0\mathrm{p}}, \mathbb{A}, \delta_{\mathrm{p}})$ is a CFSM.

A *configuration* of S:

$$s = (\vec{q}, \vec{w})$$

- $\vec{q} = (q_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$ *the overall state of the system*
  where $q_{\mathrm{p}} \in Q_{\mathrm{p}}$ *the current state of machine* M_{p}
- $\vec{w} = (w_{\mathrm{pq}})_{\mathrm{pq} \in \mathsf{Chan}}$ with $w_{\mathrm{pq}} \in \mathbb{A}^*$. *the current contents of channels*

The initial configuration of S is $s_0 = (\vec{q_0}, \vec{\varepsilon})$ with $\vec{q_0} = (q_{0_{\mathrm{p}}})_{\mathrm{p} \in \mathbf{P}}$.

Systems of CFSMs

A *system* of CFSMs:

$$S = (M_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$$

- $\mathbf{P}$ is the set of *roles* (participants) of S, and
- for each $\mathrm{p} \in \mathbf{P}$, $M_{\mathrm{p}} = (Q_{\mathrm{p}}, q_{0\mathrm{p}}, \mathbb{A}, \delta_{\mathrm{p}})$ is a CFSM.

A *configuration* of S:

$$s = (\vec{q}, \vec{w})$$

- $\vec{q} = (q_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$ *the overall state of the system*
  where $q_{\mathrm{p}} \in Q_{\mathrm{p}}$ *the current state of machine M_{p}*
- $\vec{w} = (w_{\mathrm{pq}})_{\mathrm{pq} \in \mathsf{Chan}}$ with $w_{\mathrm{pq}} \in \mathbb{A}^*$. *the current contents of channels*

The initial configuration of S is $s_0 = (\vec{q_0}, \vec{\varepsilon})$ with $\vec{q_0} = (q_{0_{\mathrm{p}}})_{\mathrm{p} \in \mathbf{P}}$.

Systems of CFSMs

A *system* of CFSMs:

$$S = (M_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$$

- $\mathbf{P}$ is the set of *roles* (participants) of S, and
- for each $\mathrm{p} \in \mathbf{P}$, $M_{\mathrm{p}} = (Q_{\mathrm{p}}, q_{0\mathrm{p}}, \mathbb{A}, \delta_{\mathrm{p}})$ is a CFSM.

A *configuration* of S:

$$s = (\vec{q}, \vec{w})$$

- $\vec{q} = (q_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$ *the overall state of the system*
  where $q_{\mathrm{p}} \in Q_{\mathrm{p}}$ *the current state of machine* M_{p}
- $\vec{w} = (w_{\mathrm{pq}})_{\mathrm{pq} \in \mathsf{Chan}}$ with $w_{\mathrm{pq}} \in \mathbb{A}^*$. *the current contents of channels*

The initial configuration of S is $s_0 = (\vec{q_0}, \vec{\varepsilon})$ with $\vec{q_0} = (q_{0_{\mathrm{p}}})_{\mathrm{p} \in \mathbf{P}}$.

Systems of CFSMs

A *system* of CFSMs:

$$S = (M_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$$

- $\mathbf{P}$ is the set of *roles* (participants) of S, and
- for each $\mathrm{p} \in \mathbf{P}$, $M_{\mathrm{p}} = (Q_{\mathrm{p}}, q_{0\mathrm{p}}, \mathbb{A}, \delta_{\mathrm{p}})$ is a CFSM.

A *configuration* of S:

$$s = (\vec{q}, \vec{w})$$

- $\vec{q} = (q_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$ *the overall state of the system*
  where $q_{\mathrm{p}} \in Q_{\mathrm{p}}$ *the current state of machine* M_{p}
- $\vec{w} = (w_{\mathrm{pq}})_{\mathrm{pq} \in \mathsf{Chan}}$ with $w_{\mathrm{pq}} \in \mathbb{A}^*$. *the current contents of channels*

The initial configuration of S is $s_0 = (\vec{q_0}, \vec{\varepsilon})$ with $\vec{q_0} = (q_{0_{\mathrm{p}}})_{\mathrm{p} \in \mathbf{P}}$.

Systems of CFSMs

A *system* of CFSMs:

$$S = (M_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$$

- $\mathbf{P}$ is the set of *roles* (participants) of S, and
- for each $\mathrm{p} \in \mathbf{P}$, $M_{\mathrm{p}} = (Q_{\mathrm{p}}, q_{0\mathrm{p}}, \mathbb{A}, \delta_{\mathrm{p}})$ is a CFSM.

A *configuration* of S:

$$s = (\vec{q}, \vec{w})$$

- $\vec{q} = (q_{\mathrm{p}})_{\mathrm{p} \in \mathbf{P}}$ *the overall state of the system*
  where $q_{\mathrm{p}} \in Q_{\mathrm{p}}$ *the current state of machine* M_{p}
- $\vec{w} = (w_{\mathrm{pq}})_{\mathrm{pq} \in \mathsf{Chan}}$ with $w_{\mathrm{pq}} \in \mathbb{A}^*$. *the current contents of channels*

The initial configuration of S is $s_0 = (\vec{q_0}, \vec{\varepsilon})$ with $\vec{q_0} = (q_{0_{\mathrm{p}}})_{\mathrm{p} \in \mathbf{P}}$.

System transitions:

$$(q, w) \xrightarrow{\texttt{AB}!msg} (q', w')$$

- $(q_{\texttt{A}},\ \texttt{AB}!msg,\ q'_{\texttt{A}}) \in \delta_{\texttt{A}}$
- $\forall \texttt{p} \neq \texttt{A}.\ q'_{\texttt{p}} = q_{\texttt{p}}$
- $w'_{\texttt{AB}} = w_{\texttt{AB}} \cdot msg$ and $\forall \texttt{pr} \neq \texttt{AB}.\ w'_{\texttt{pr}} = w_{\texttt{pr}}$

Similarly for

$$(q, w) \xrightarrow{\texttt{AB}?msg} (q', w')$$

Synchronous communications

It is easy to equip CFSMs also with a synchronous communications.

Choreographies for CFSMs systems: Which description formalism?

It takes a thief to catch a thief... so

Choreography Automata

Choreographies for CFSMs systems: Which description formalism?

It takes a thief to catch a thief... so

Choreography Automata

Choreographies for CFSMs systems: Which description formalism?

It takes a thief to catch a thief... so

Choreography Automata

Choreographies for CFSMs systems: Which description formalism?

It takes a thief to catch a thief... so

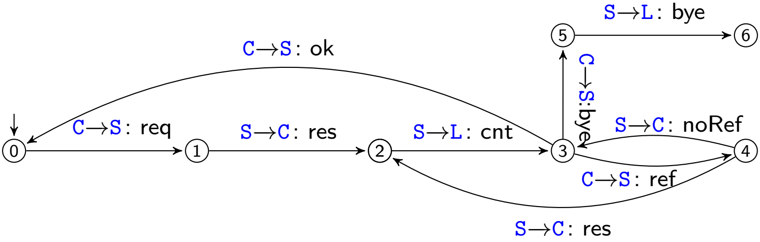
Choreography Automata

Choreographies for CFSMs systems: Which description formalism?

It takes a thief to catch a thief... so

Choreography Automata

Choreography Automata through an Example



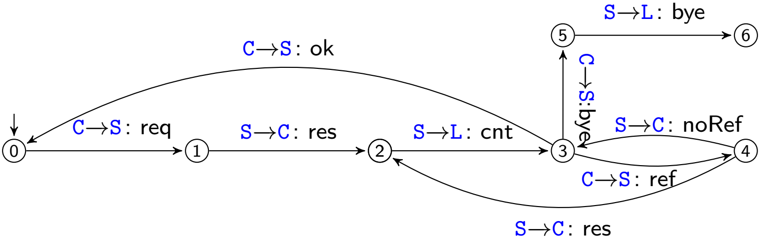
An apparent resemblance

Choreography Automata **vs.** Conversation Protocols
(by Bultan et al.)

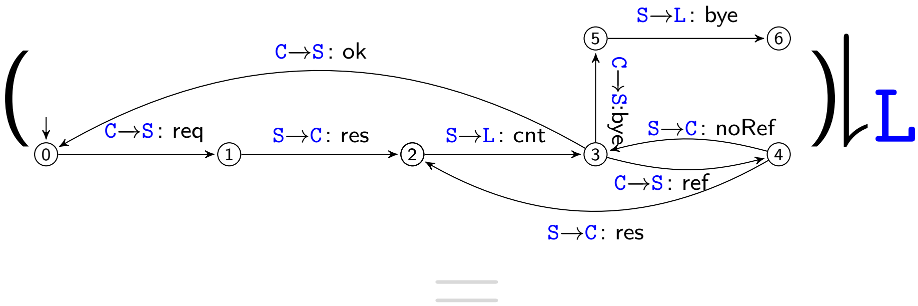
They look alike, but actually their semantics and underlying communication models do differ.

(a thorough comparison in the Related Works section of the paper)

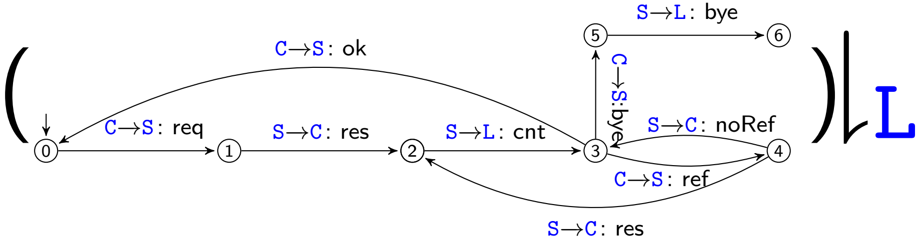
Choreography Automata through an Example



Projection


C→S: ok
S→L: bye
C→S:bye
S→C: noRef
C→S: req
S→C: res
S→L: cnt
C→S: ref
S→C: res
0
1
2
3
4
5
6
L

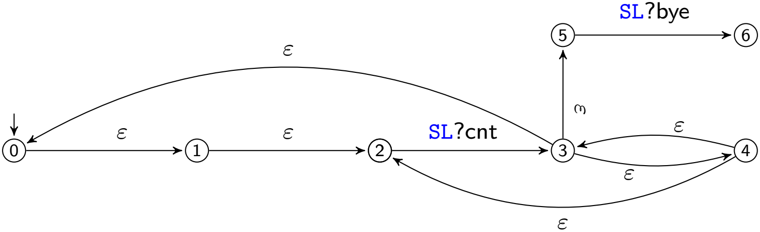
Projection



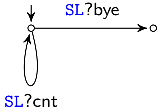
S→L: bye
C→S: ok
C→S:bye
C→S: req
S→C: res
S→L: cnt
S→C: noRef
C→S: ref
S→C: res
0
1
2
3
4
5
6
L

=

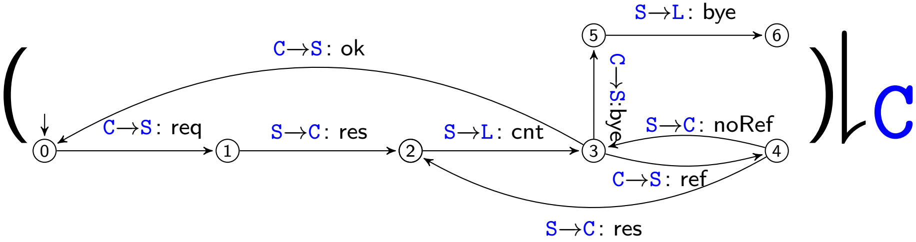
Projection



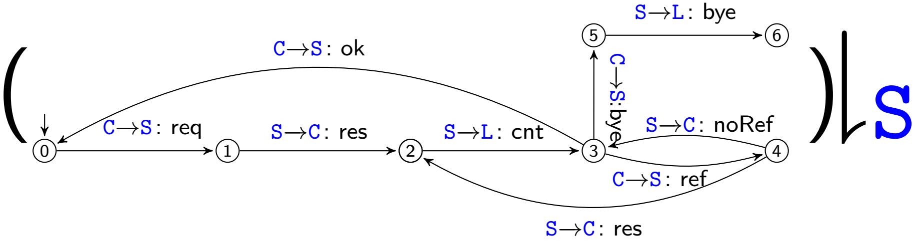
Projection



Projection

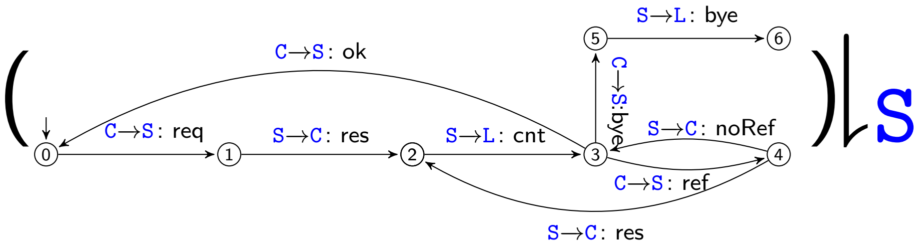

S→L: bye
C→S: ok
C→S:bye
S→C: noRef
C→S: req
S→C: res
S→L: cnt
0
1
2
3
4
5
6
C→S: ref
S→C: res
C

Projection



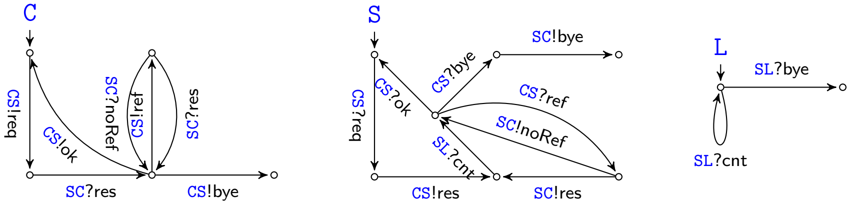
C→S: ok
S→L: bye
C→S: req
S→C: res
S→L: cnt
C→S:bye
S→C: noRef
C→S: ref
S→C: res
0
1
2
3
4
5
6
S

Projection



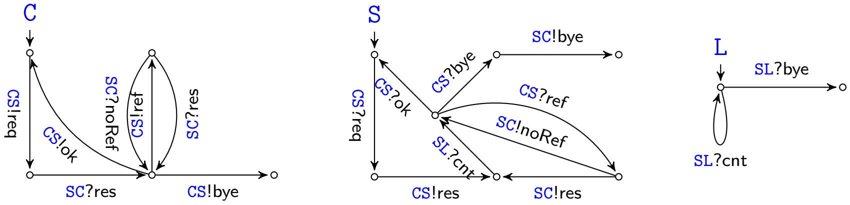
S→L: bye
C→S: ok
C→S:bye
S→C: noRef
C→S: req
S→C: res
S→L: cnt
C→S: ref
S→C: res
0
1
2
3
4
5
6
S

Projection



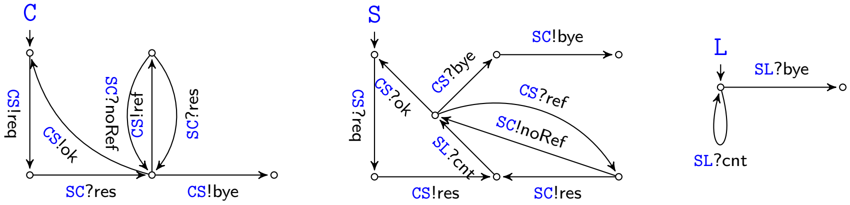
- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free** *i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)*

Projection



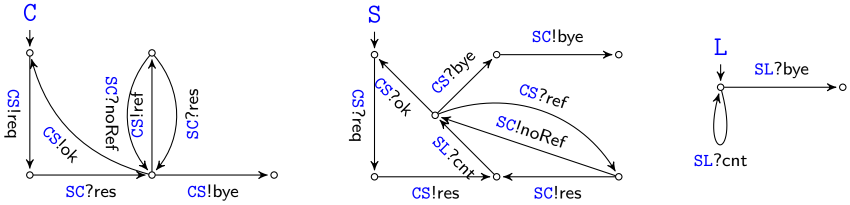
- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free** *i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)*

Projection



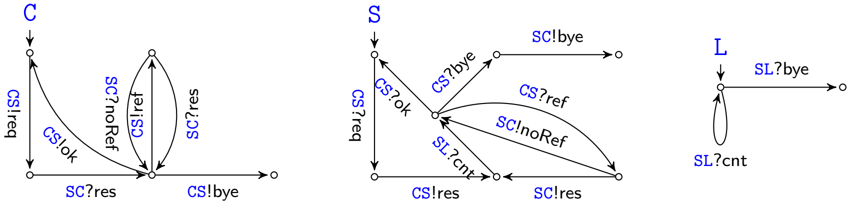
- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free**
  i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)

Projection



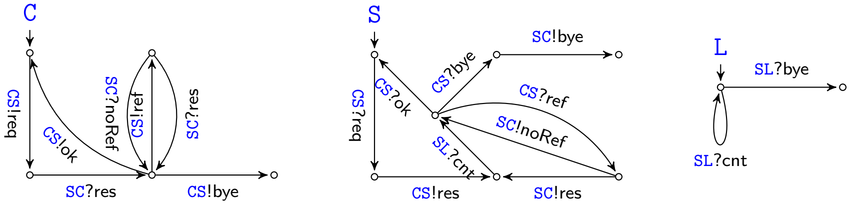
- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free**
  i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)

Projection



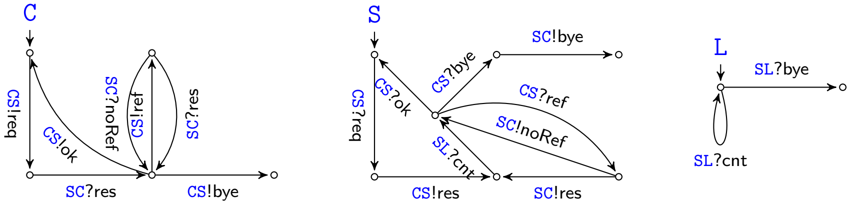
- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free** *i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)*

Projection



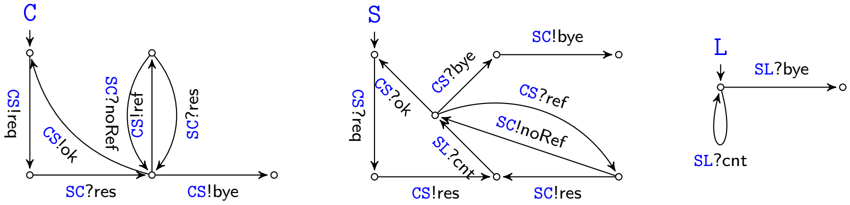
- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free** *i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)*

Projection



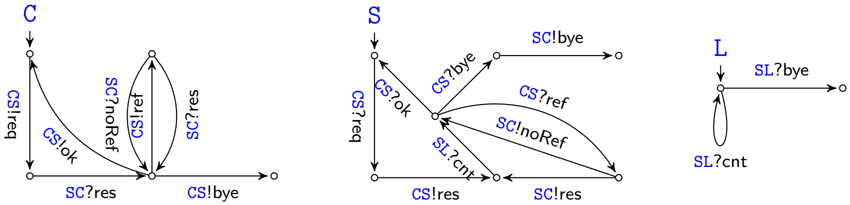
- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free**
  i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)

Projection



- The behaviour of the system of CFSMs perfectly match the overall behaviour described by the choreography automata:
- The system is **Live**, *i.e. if a machine is willing to perform some actions, the system can evolve so that one eventually is done*
- The system is **Deadlock-Free** *i.e. it will never get stuck (the system does progress)*
- The system is **Lock-Free**
  i.e. if a machine can perform some actions, sooner or later it will do one (any single machine does progress)

Projection



Both for **Synchronous** and **Asynchronous** communications

There ain't no such thing as a free lunch

Only the projections of *well-behaved* Choreography Automata are *well-behaved*.

Theorem

Given a well-formed c-automaton CA, the system obtained by projection, $(CA\lfloor_{\mathtt{A}})_{\mathtt{A}\in\mathcal{P}}$, is live, lock-free, and deadlock-free both for synchronous and asynchronous communications.

Definition (Well-formedness)

A c-automaton CA is well-formed if (roughly)

- when there is a choice, a single participant decides;
- all the partecipants are eventuelly made aware of the choices made;
- parallelism of independent interactions must be made explicit by interleaving them

There ain't no such thing as a free lunch

Only the projections of *well-behaved* Choreography Automata are *well-behaved*.

Theorem

Given a well-formed c-automaton CA, the system obtained by projection, $(CA\lfloor_{\mathtt{A}})_{\mathtt{A}\in\mathcal{P}}$, *is live, lock-free, and deadlock-free both for synchronous and asynchronous communications.*

Definition (Well-formedness)

A c-automaton CA is well-formed if (roughly)

- when there is a choice, a single participant decides;
- all the partecipants are eventuelly made aware of the choices made;
- parallelism of independent interactions must be made explicit by interleaving them

There ain't no such thing as a free lunch

Only the projections of *well-behaved* Choreography Automata are *well-behaved*.

Theorem

Given a well-formed c-automaton CA, the system obtained by projection, $(CA\downharpoonright_{\mathtt{A}})_{\mathtt{A}\in\mathcal{P}}$, is live, lock-free, and deadlock-free both for synchronous and asynchronous communications.

Definition (Well-formedness)

A c-automaton CA is well-formed if (roughly)

- when there is a choice, a single participant decides;
- all the partecipants are eventuelly made aware of the choices made;
- parallelism of independent interactions must be made explicit by interleaving them

There ain't no such thing as a free lunch

Only the projections of *well-behaved* Choreography Automata are *well-behaved*.

Theorem

Given a well-formed c-automaton CA, the system obtained by projection, $(CA\lfloor_{\mathtt{A}})_{\mathtt{A} \in \mathcal{P}}$*, is live, lock-free, and deadlock-free both for synchronous and asynchronous communications.*

Definition (Well-formedness)

A c-automaton CA is well-formed if (roughly)

- when there is a choice, a single participant decides;
- all the partecipants are eventuelly made aware of the choices made;
- parallelism of independent interactions must be made explicit by interleaving them

There ain't no such thing as a free lunch

Only the projections of *well-behaved* Choreography Automata are *well-behaved*.

Theorem

Given a well-formed c-automaton CA, the system obtained by projection, $(CA{\downharpoonleft}_{\mathtt{A}})_{\mathtt{A}\in\mathcal{P}}$*, is live, lock-free, and deadlock-free both for synchronous and asynchronous communications.*

Definition (Well-formedness)

A c-automaton CA is well-formed if (roughly)

- when there is a choice, a single participant decides;
- all the partecipants are eventuelly made aware of the choices made;
- parallelism of independent interactions must be made explicit by interleaving them

There ain't no such thing as a free lunch

Only the projections of *well-behaved* Choreography Automata are *well-behaved*.

Theorem

Given a well-formed c-automaton CA, the system obtained by projection, $(CA\restriction_{\mathtt{A}})_{\mathtt{A}\in\mathcal{P}}$, is live, lock-free, and deadlock-free both for synchronous and asynchronous communications.

Definition (Well-formedness)

A c-automaton CA is well-formed if (roughly)

- when there is a choice, a single participant decides;
- all the partecipants are eventuelly made aware of the choices made;
- parallelism of independent interactions must be made explicit by interleaving them

A promising future development

Usually choreographic models are good for the description of **closed** systems. What about **open** systems? The

"participants as interfaces" approach to choreography for open systems.

One of the main motivations to develop a choreography model based on automata was to have a formalism enabling to internally describe a composition mechanism of global specifi cations (preserving well-formedness)

A promising future development

Usually choreographic models are good for the description of **closed** systems. What about **open** systems? The

“participants as interfaces” approach to choreography for open systems.

One of the main motivations to develop a choreography model based on automata was to have a formalism enabling to internally describe a composition mechanism of global specifi cations (preserving well-formedness)

A promising future development

Usually choreographic models are good for the description of **closed** systems. What about **open** systems? The

"participants as interfaces" approach to choreography for open systems.

One of the main motivations to develop a choreography model based on automata was to have a formalism enabling to internally describe a composition mechanism of global specifi cations (preserving well-formedness)

A promising future development

Usually choreographic models are good for the description of **closed** systems. What about **open** systems? The

“participants as interfaces” approach to choreography for open systems.

One of the main motivations to develop a choreography model based on automata was to have a formalism enabling to internally describe a composition mechanism of global specifi cations (preserving well-formedness)

A promising future development

Usually choreographic models are good for the description of **closed** systems. What about **open** systems? The

"participants as interfaces" approach to choreography for open systems.

One of the main motivations to develop a choreography model based on automata was to have a formalism enabling to internally describe a composition mechanism of global specifi cations (preserving well-formedness)