Lecture 4 : advanced

Multiparty Session Types

BETTY Summer School 2016

Laura Bocchi, University of Kent, l.bocchi@kent.ac.uk

Agenda

- MPSTs extensions
- Design by Contract
- Beyond static typing
- Relationship with automata
- Time
- Realizability

MPSTs extensions

- Structure of interactions
  - Symmetric sum types
    [Nielsen,Yoshida,Honda@Express'10][Henriksen et al.@FHIES'12]
  - Exceptions/Escapes
    [Capecchi,Giachino,Yoshida@FTTCS'10][Carbone,Yoshida,Honda@SFM'09]
  - Dynamic multirole session types
    [Denielou,Yoshida@POPL'11]
  - Parameterised sessions
    [Denielou et al.@LMCS'12]
  - Nested sessions
    [Demangeon,Honda@CONCUR'12]
- Design by Contract (assertions on message content)
  [Bocchi,Honda,Tuosto,Yoshida@CONCUR'10] [Bocchi,Demangeon,Yoshida@TGC'12]
- Time [Bocchi,Yang,Yoshida@CONCUR'14]

Design by Contract

Assertions = Types + Logical Formulae

- Type signature

```
int hello(int i)
```

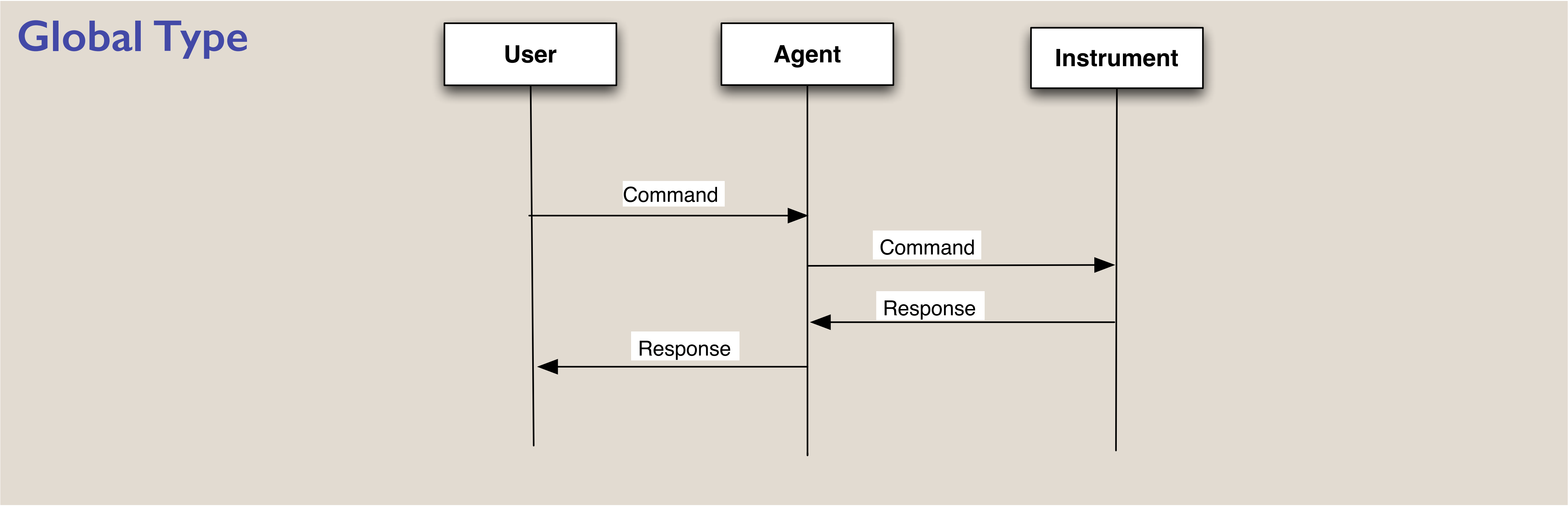
- Assertion

```
int hello(int i){
        pre: {i>10}
        post: {result>0}
}
```

- Building systems on the basis of precise contracts
  - restrain defensive programming
  - provide robustness

[Bertrand Meyer,1992]

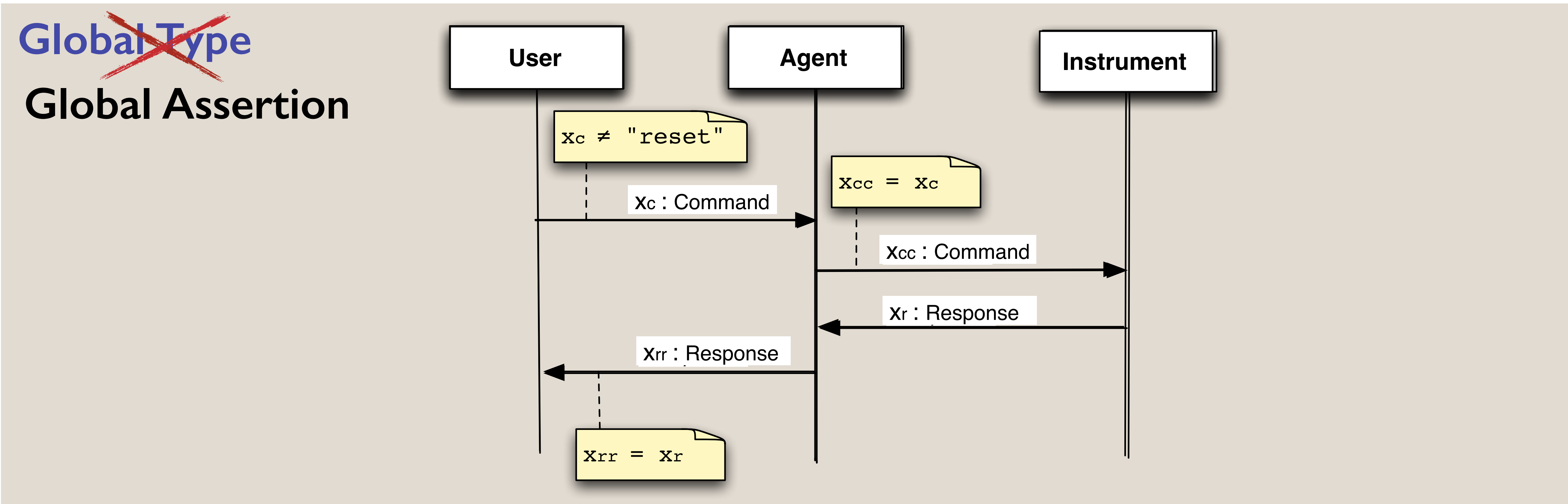
A theory of DbC for concurrency



$\texttt{User} \rightarrow \texttt{Agent} : s_1(\texttt{Command}).$
$\texttt{Agent} \rightarrow \texttt{Instrument} : s_2(\texttt{Command}).$
$\texttt{Instrument} \rightarrow \texttt{Agent} : s_1(\texttt{Response}).$
$\texttt{Agent} \rightarrow \texttt{User} : s_3(\texttt{Response})$

Bocchi,Honda,Tuosto,Yoshida@CONCUR'10
"A theory of Design by Contract for Distributed Multiparty Interactions"

A theory of DbC for concurrency

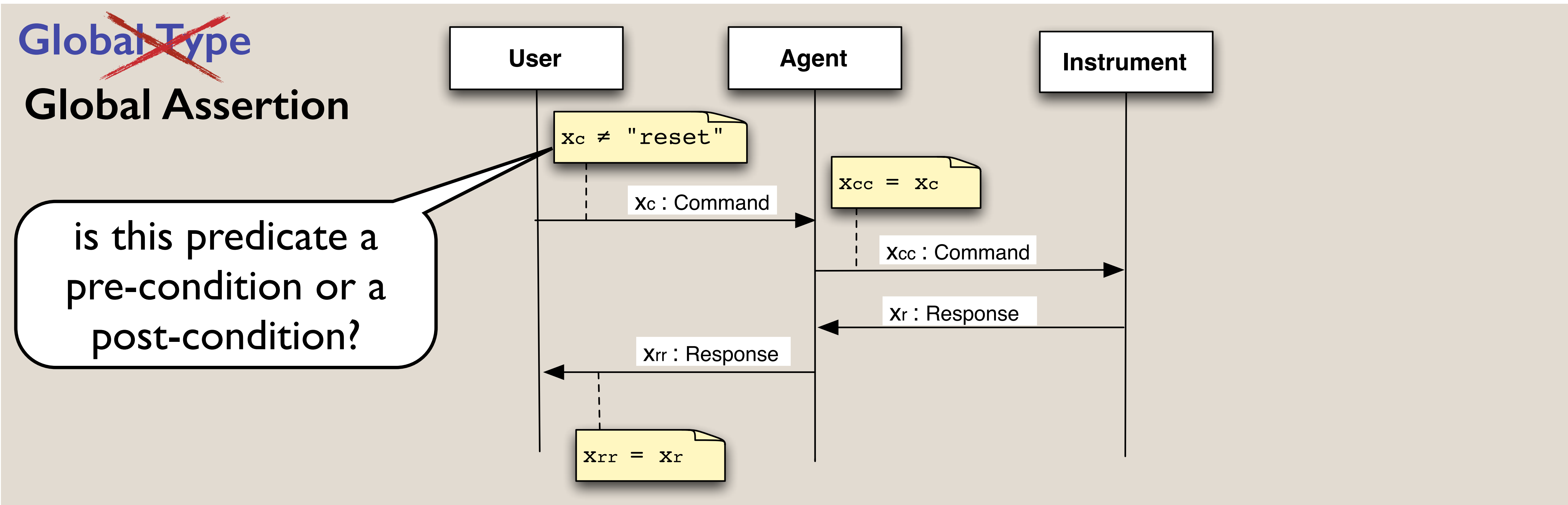


$$\texttt{User} \to \texttt{Agent} : s_1(x_c : \texttt{Command})\{x_c \neq reset\}.$$
$$\texttt{Agent} \to \texttt{Instrument} : s_2(x_{cc} : \texttt{Command})\{x_{cc} = x_c\}.$$
$$\texttt{Instrument} \to \texttt{Agent} : s_1(x_r : \texttt{Response}).$$
$$\texttt{Agent} \to \texttt{User} : s_3(x_{rr} : \texttt{Response})\{x_{rr} = x_r\}$$

Bocchi,Honda,Tuosto,Yoshida@CONCUR'10

"A theory of Design by Contract for Distributed Multiparty Interactions"

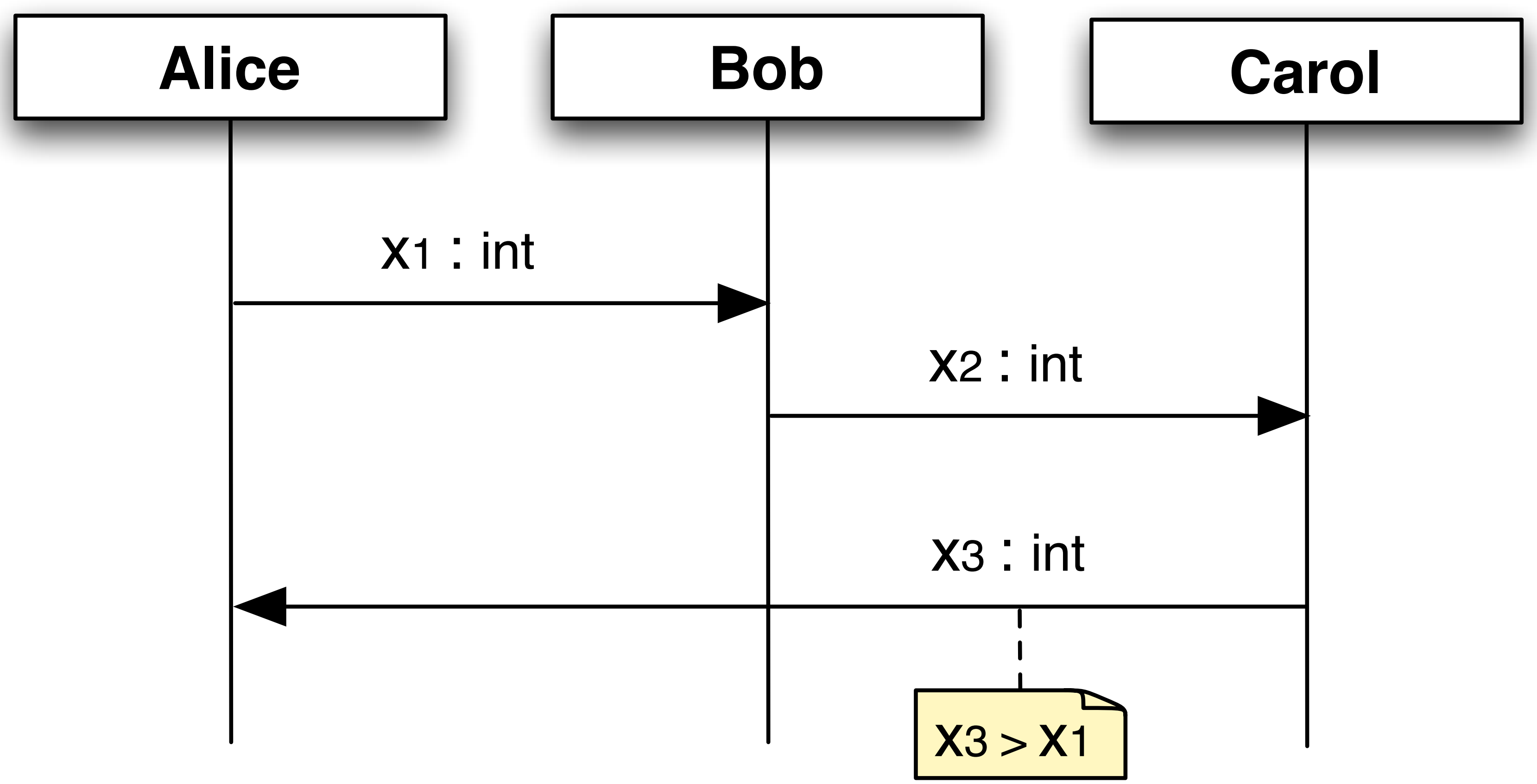
A theory of DbC for concurrency



- Participants have different views of the contract
- Responsibilities spread among the participants
- e.g., a predicate associated to an interaction is
  - a pre-condition for the receiver
  - a post-condition for the sender

Rethinking realizability

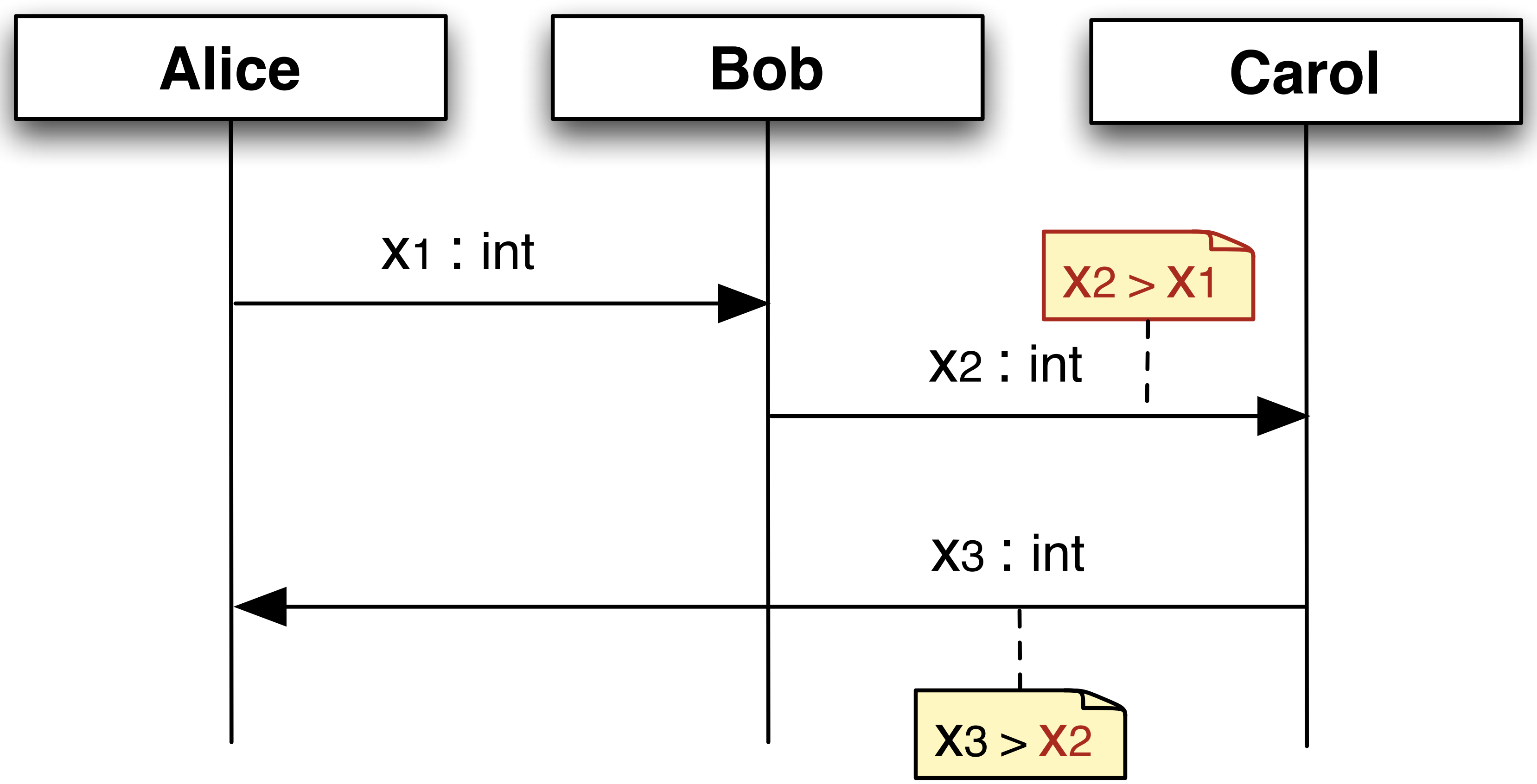
- Is this a good choreography?



- No, Carol does not have enough information to fulfil her obligation

Rethinking realizability

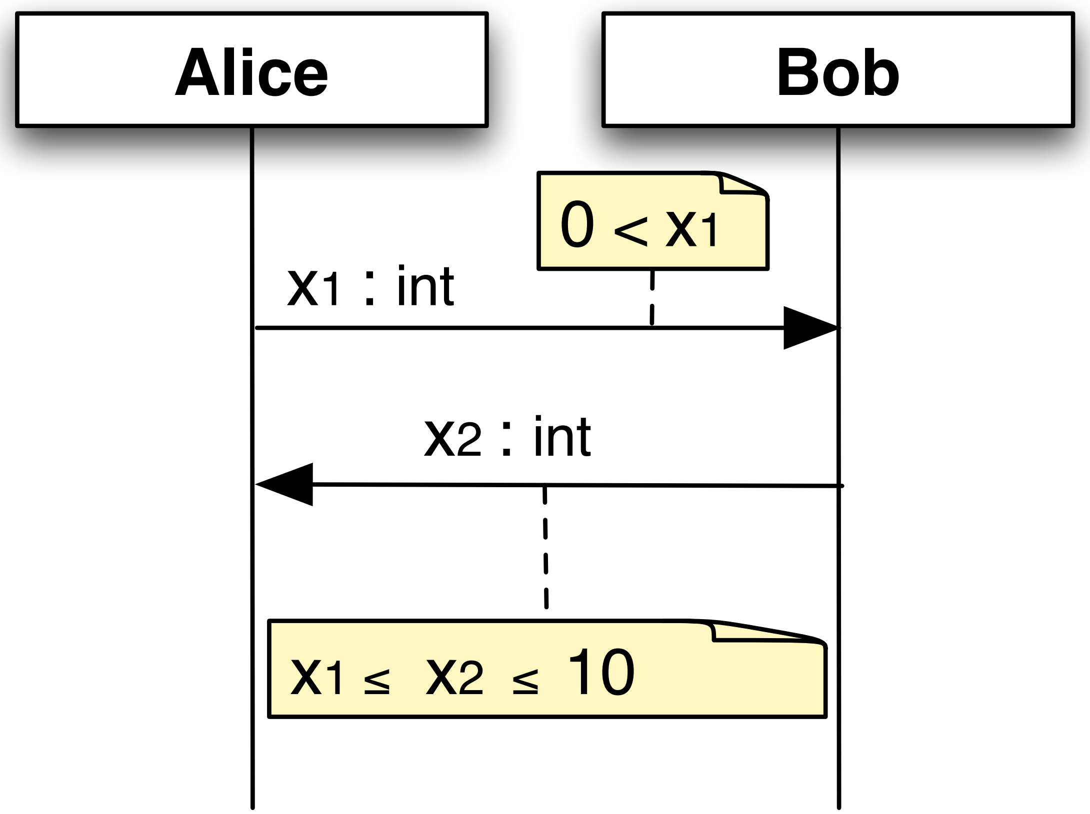
- Is this a good choreography?



- Yes, Carol knows the variables in the predicate she must guarantee

Rethinking realizability

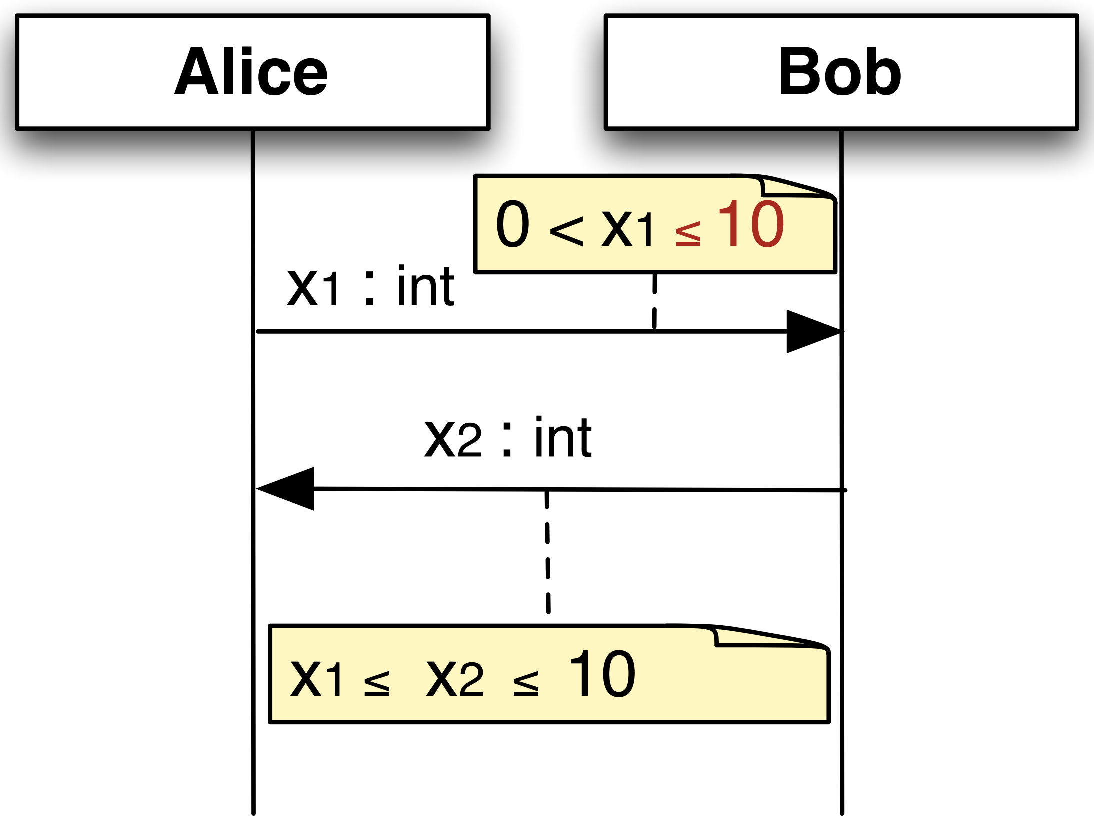
- Is this a good choreography?



- No, Bob may “find” his obligation unsatisfiable

Rethinking realizability

- Is this a good choreography?

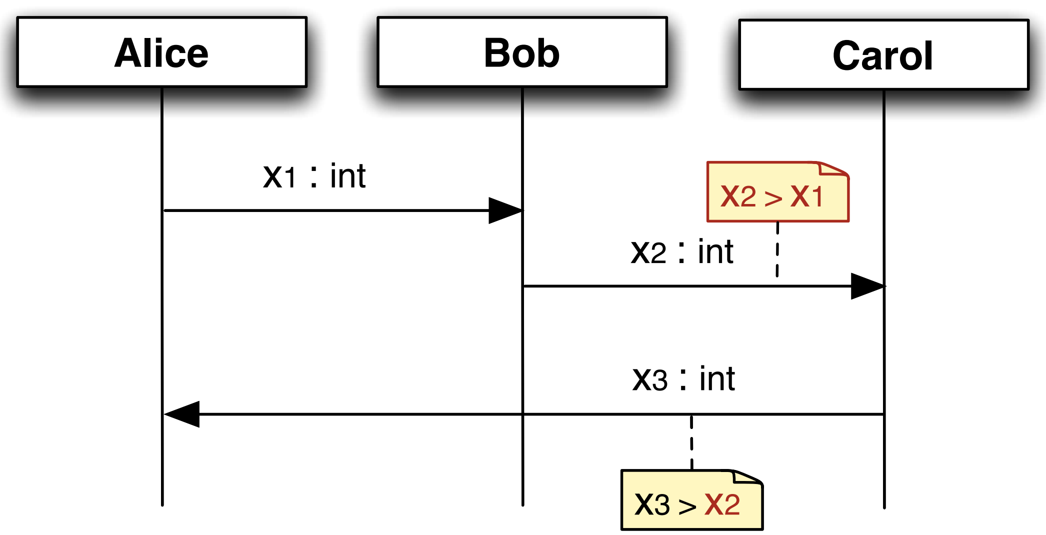


- Yes, there is always a solution to the constraint Bob must guarantee

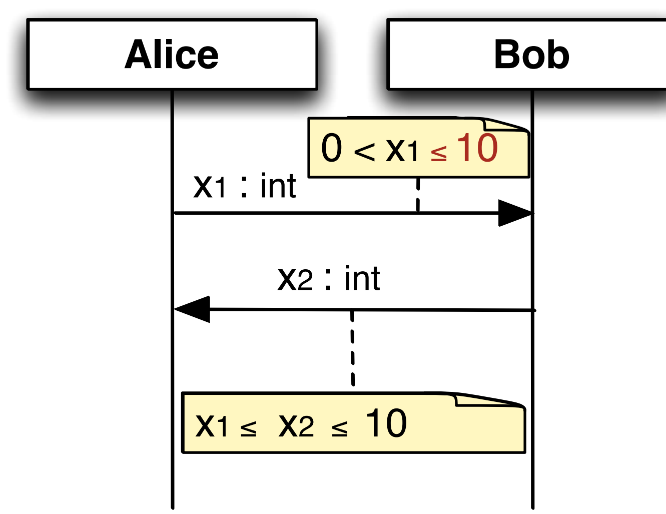
Rethinking realizability

- We want to prevent:
  - that a participant does not have enough information to make correct choices
  - that a participant has no other alternatives than violating the contract

History Sensitivity:
"a predicate only contains variables that are known to the sender"



Temporal Satisfiability:
"a process can find a valid forward path at each interaction point"



Similar to *feasibility* in
[Apt, Francez & Katz, POPL'87]

Bocchi,Lange,Tuosto@ICE'11,SACS'12
Amending Contracts for Choreographies

Projecting assertions

- Projection - build the strongest set of preconditions for each role

$$\texttt{Seller} \to \texttt{Buyer}: k_1\ (cost : \mathsf{Int})\{cost > 10\}.\texttt{Buyer} \to \texttt{Bank}: k_2\ (pay : \mathsf{Int})\{pay \geqslant cost\}.\mathsf{end}$$

$$?(\texttt{Buyer},\ (pay : \mathsf{Int}))\{pay \geq cost\}.\texttt{end}$$

$$\exists cost.cost > 10 \wedge pay \geq cost$$

Typing (some ideas…)

typing environment

assertion environment $\quad \mathcal{C}\,;\Gamma \vdash P \;:>\; \Delta \quad$ session environment

$$\frac{\Gamma \vdash e : S \qquad \mathcal{C} \supset A\left[{}^{e}/_{x}\right] \text{ valid} \qquad \mathcal{C}\,;\Gamma \vdash P\left[{}^{e}/_{x}\right] \;:>\; \Delta, c : \mathcal{T}\left[{}^{e}/_{x}\right]}{\mathcal{C}\,;\Gamma \vdash c[\mathsf{p}]\,!e;P \;:>\; \Delta, c : !\,\mathsf{p}\,(x\!: S)\{A\};\mathcal{T}}\;[\text{SND}]$$

$$\frac{\mathcal{C} \wedge A\,;\Gamma, x : S \vdash P \;:>\; \Delta,\, c : \mathcal{T}}{\mathcal{C}\,;\Gamma \vdash c[\mathsf{p}]?(x);P \;:>\; \Delta, c : ?\,\mathsf{p}\,(x\!: S)\{A\};\mathcal{T}}\;[\text{RCV}]$$

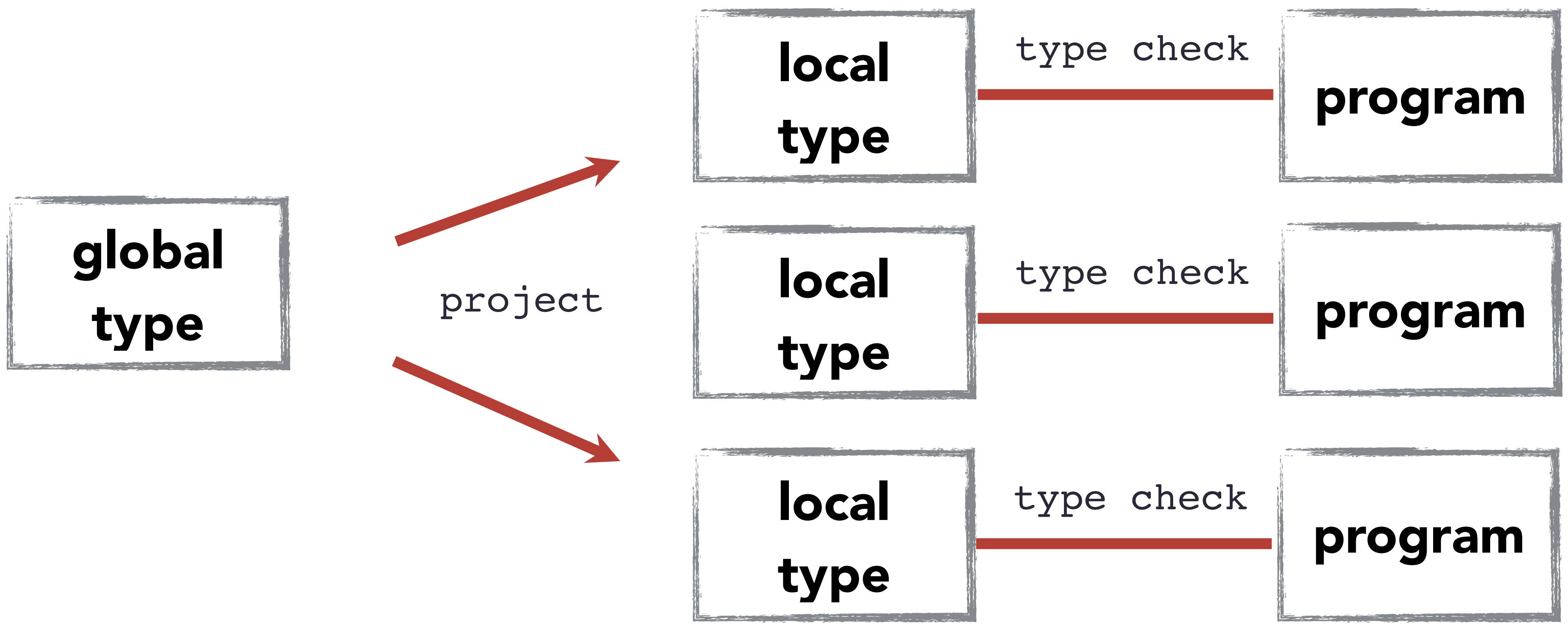
Properties

- **Decidability** of type-checking depends on the logics (e.g., some fragment of Presburger arithmetic)

- **Soundness** : the behaviour of the system can be mimicked by the types

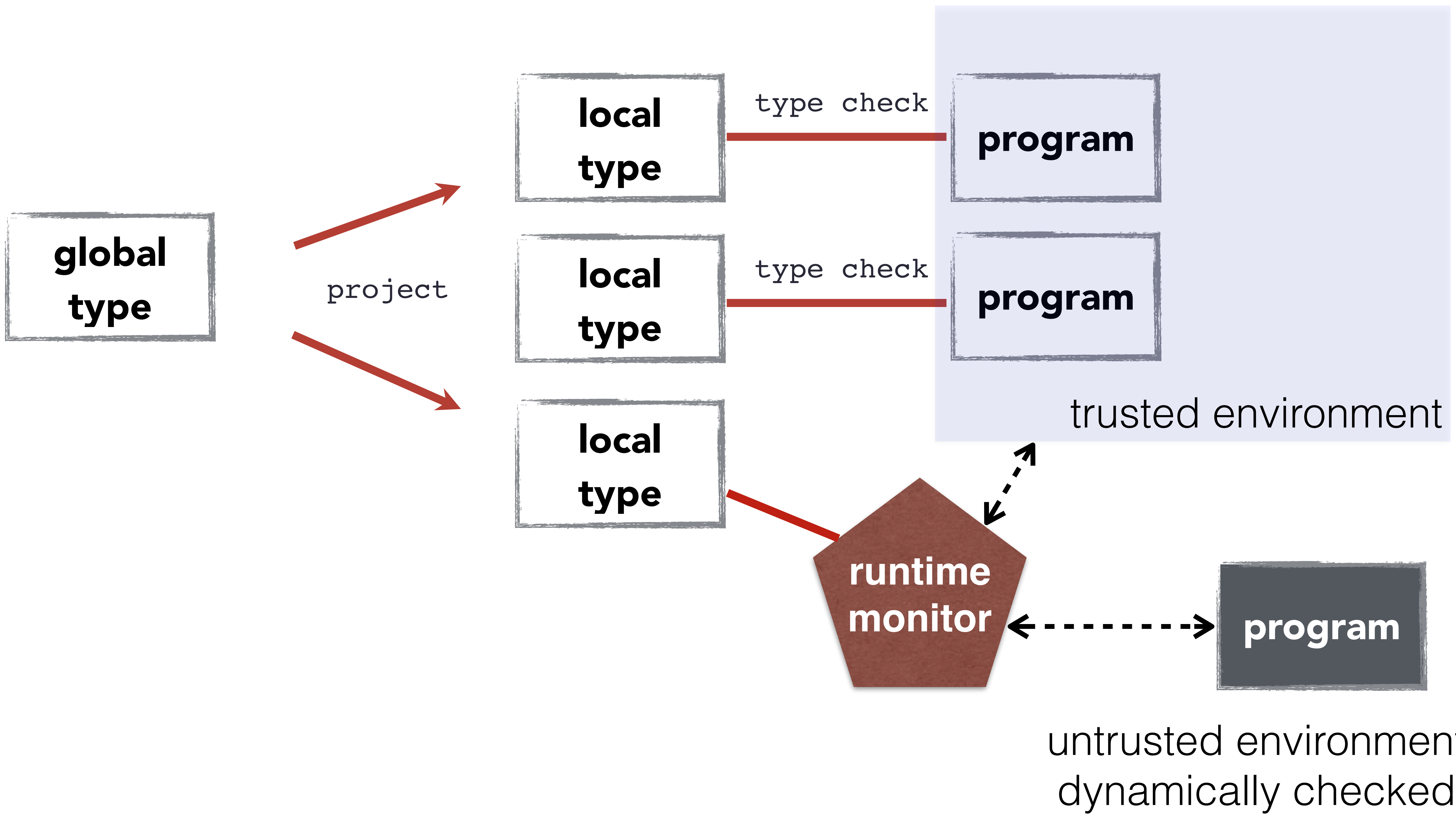
- **Completeness** : if a process is well-behaved then it can be typed (for a class of processes that do not “get stuck” at a session initiation)

Static typing



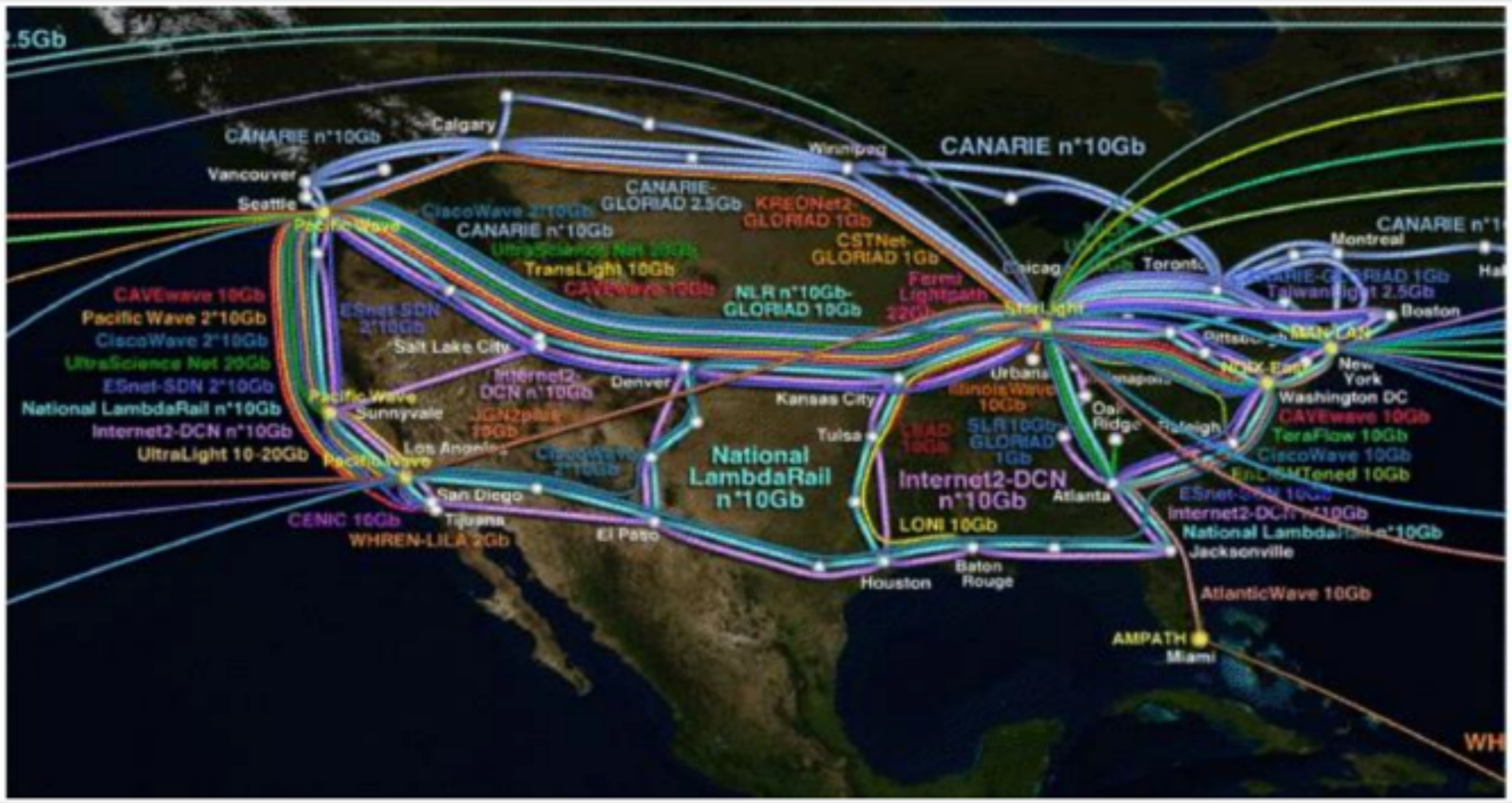
If all programs are well typed then the system is well behaved

Static typing



Ocean Observatory Initiative (OOI)

OOI aims: to deploy an infrastructure (global network) to expand the scientists' ability to remotely study the ocean

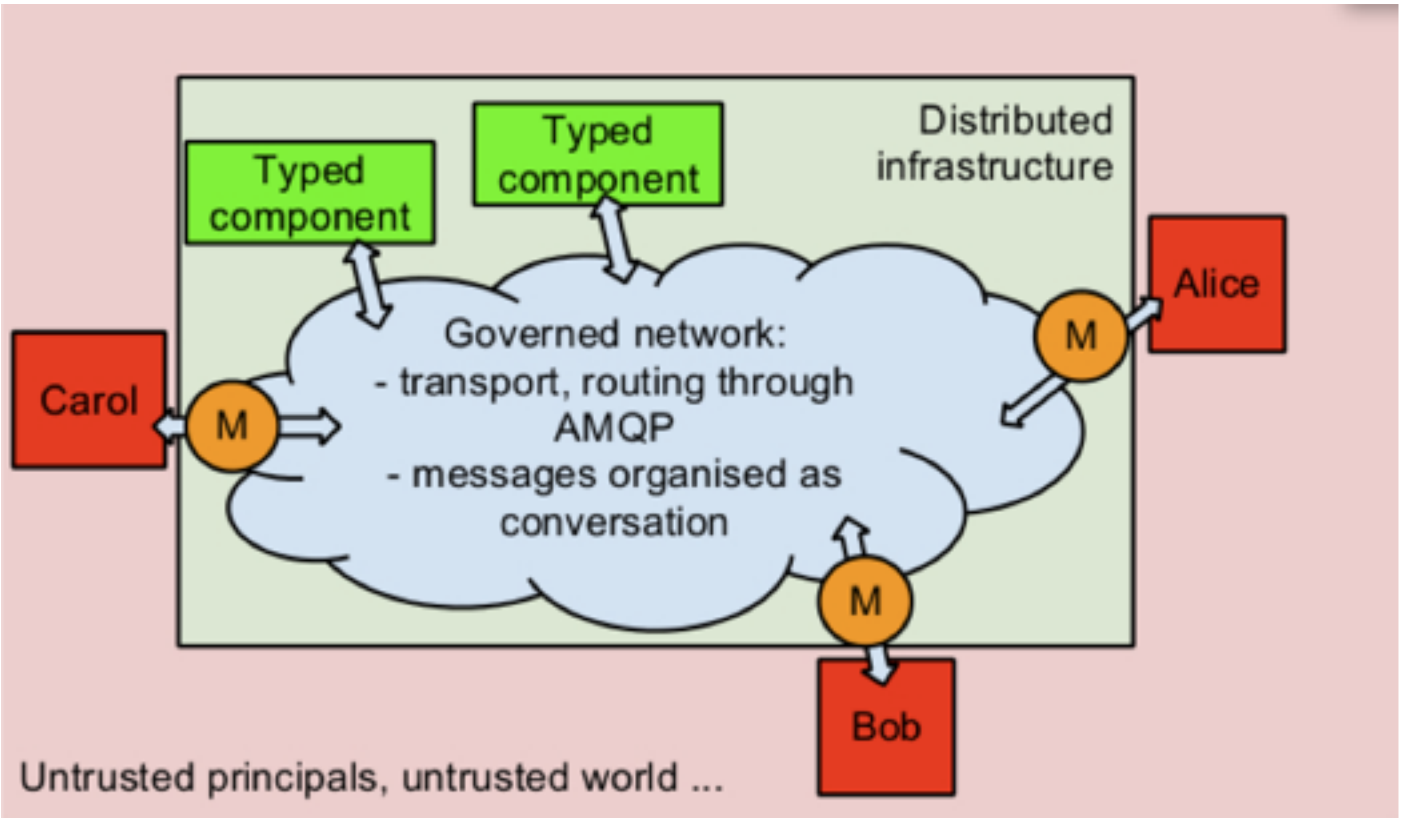


Usage: Integrate real-time data acquisition, processing and data storage for ocean research,…

OOI: verification challenges

- applications written in **different** languages, running on **heterogeneous** hardware in an **asynchronous** network.
- different authentication domains, external **untrusted** applications
- various distributed protocols
- requires correct, safe interactions

Monitoring with MPST



- Monitors between trusted network and untrusted apps
  - drop violating incoming/outgoing messages
  - ensure interoperability (no access to source code)

Bocchi,Chen,Demangeon,Honda,Yoshida@FORTE'13
Monitoring Networks through Multiparty Session Types

Run-time monitoring

$$N = [P_1]_\alpha \mid [P_2]_\beta \mid \ldots \mid \mathsf{M}_\alpha \mid \mathsf{M}_\beta \mid \ldots \mid \langle r, h \rangle$$

$\alpha : \langle \Gamma, \Delta \rangle$ routing information

specification (with assertions)

Satisfaction $\models N : \Sigma$

- if Σ expects an **input** then N should be able to process it
- if N performs an **output** then Σ should allow it
- still holds after reduction

Properties (1/2)

Safety

$$\models [P]_\alpha \mid \mathsf{M} \;:\; \alpha : \langle \Gamma, \Delta \rangle \text{ with } \mathsf{M} = \alpha : \langle \Gamma, \Delta \rangle.$$

a monitored process satisfies its specification

....

If N is fully monitored w.r.t. Σ then $\models N : \Sigma$.

a monitored network satisfies its (global) specifications

Properties (2/2)

Transparency

a well-behaved monitored process (resp. network)
remains so when monitored

- It allows mixed network with statically checked processes
- Extended with time [Neykova,Bocchi,Yoshida@BEAT'14]
- Transparency may not hold in the timed scenario…why?

Agenda

- MPSTs extensions
- Design by Contract
- Beyond static typing
- Relationship with automata
- Time
- Realizability

MPST & CFSM

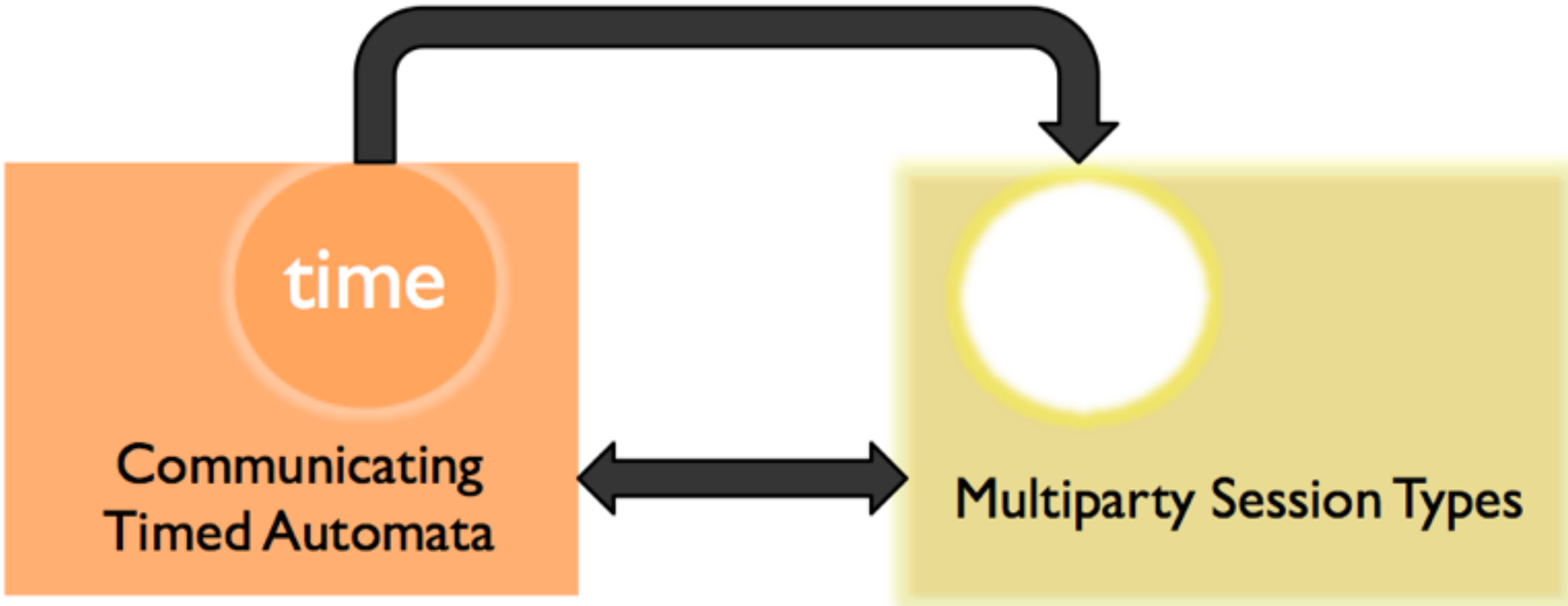
Theorem 4.3 (soundness and completeness). *Suppose S is basic and multiparty compatible. Then there exists G such that $S \approx G$. Conversely, if G is well-formed, then there exists a basic and multiparty compatible system S such that $S \approx G$.*

basic = deterministic, directed, no mixed state

multiparty compatible = in all reachable stable states, all possible (I/O) action can be matched with a complementary (O/I) action of the rest of the system after some 1-bounded executions

Deniélou,Yoshida@ICALP'13
"*Multiparty Compatibility in Communicating Automata: Characterisation and Synthesis of Global Session Types.*"

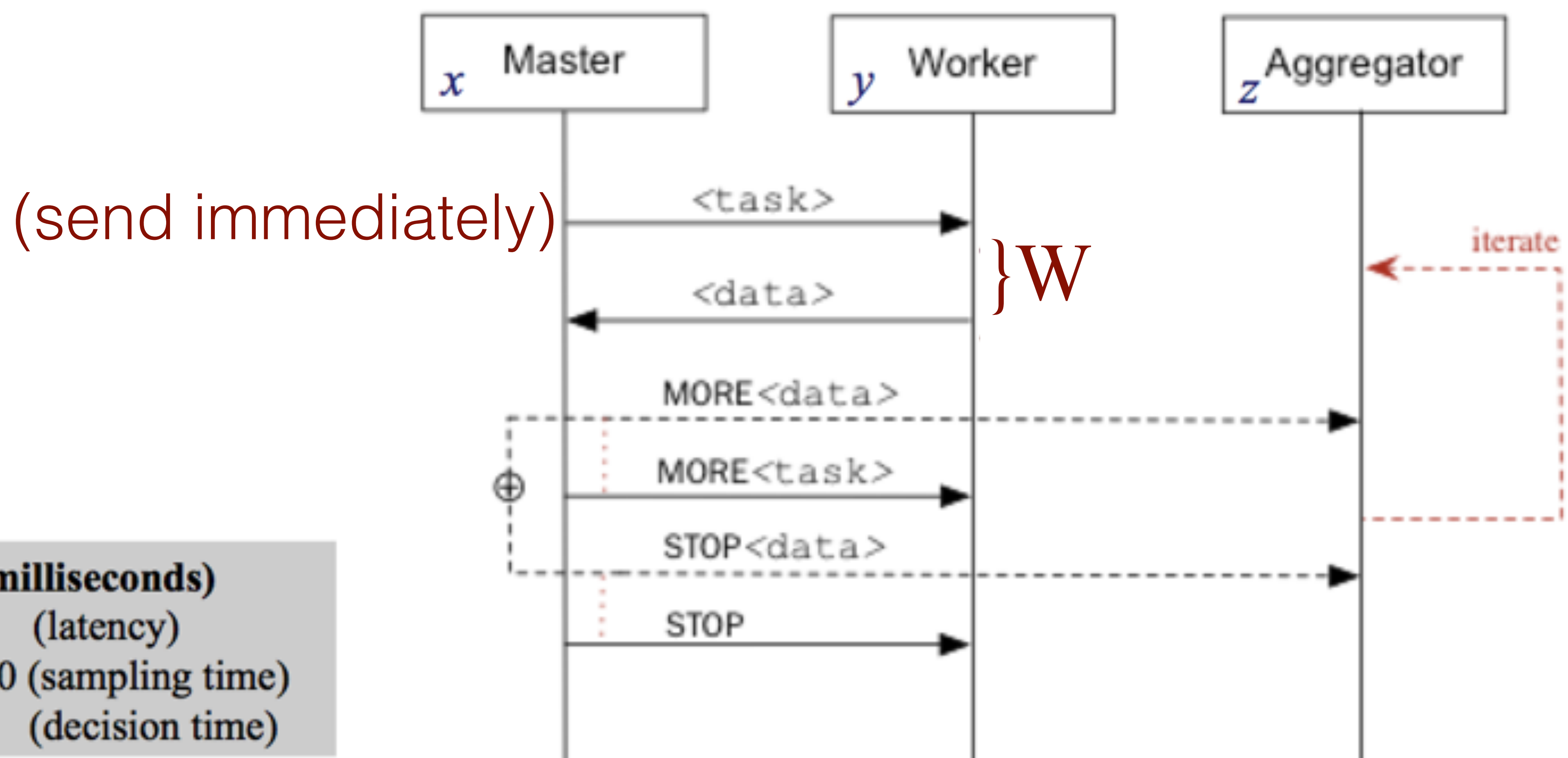
MPST, Automata & time



Timed Multiparty Session Types
Bocchi, Yang, Yoshida@CONCUR'14

Time-sensitive choreographies

- **Protocol specification**: deadlines, timeouts, repeated constraints, …
- **Web Services**: *"Reconnect no more than twice every four minutes …"* [Twitter Streaming API]
- **Sensor Networks** (on busy waiting): *"Main sources of energy inefficency in Sensor Networks are collisions and listening on idle channels"* [Ye, Heidemann & Estrin, 2002]



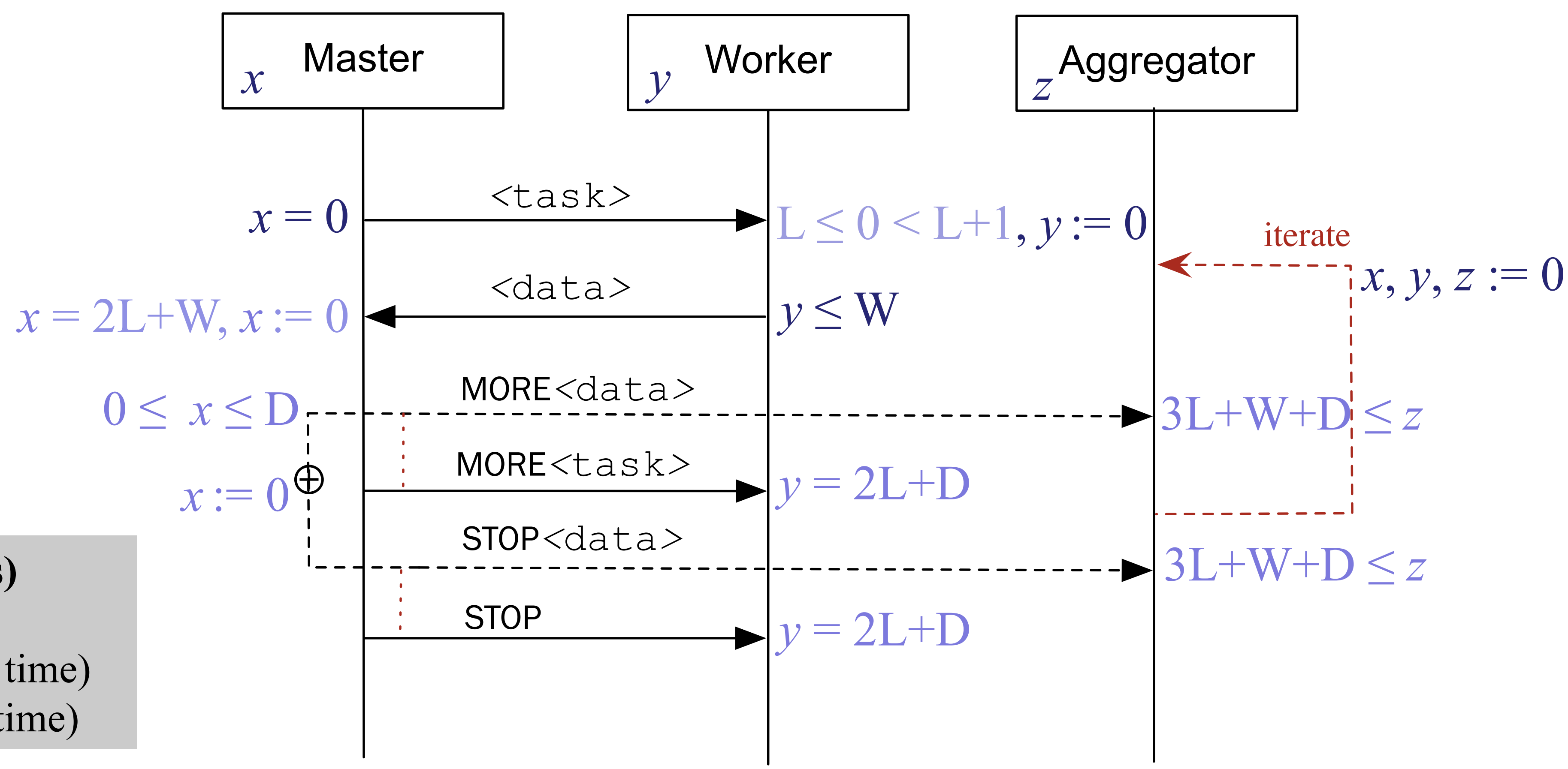
Delays (in milliseconds)
L = 400 (latency)
W = 300,000 (sampling time)
D = 2000 (decision time)

OOI OCEAN OBSERVATORIES INITIATIVE

Cognizant

Time model

- Each role owns a local clock
- All clocks synchronised at the beginning of the session
- Thereafter time flows at the same pace for all participants
- Send/receive actions guarded by constraints on clocks
- Clocks can be reset when sending or receiving



Delays (in milliseconds)
L = 400 (latency)
W = 300,000 (sampling time)
D = 2000 (decision time)

OCEAN OBSERVATORIES INITIATIVE

Cognizant

Timed MPST

$$\mathtt{G} \ ::= \ \mathtt{p} \to \mathtt{q} : \{ l_i \langle S_i \rangle \{ A_i, A_i' \} . \mathtt{G}_i \}_{i \in I} \ \mid \ \mu \mathtt{t}.\mathtt{G} \ \mid \ \mathtt{t} \ \mid \ \mathtt{end}$$

$$
\begin{array}{llll}
\mu\mathtt{t}. & \mathtt{M} \to \mathtt{W} : \langle \mathtt{task} \rangle & \{x = 0, \quad \emptyset, & L \le y \le L+1, \quad y\}. \\
& \mathtt{W} \to \mathtt{M} : \langle \mathtt{data} \rangle & \{y \le W, \quad \emptyset, & x = 2L + W, \quad x\}. \\
& \mathtt{M} \to \mathtt{A} : \{ \mathsf{MORE} \langle \mathtt{data} \rangle & \{x \le D, \quad \emptyset, & z \ge 3L + W + D, \quad z\}. \\
& \qquad \mathtt{M} \to \mathtt{W} : \mathsf{MORE} \langle \mathtt{task} \rangle & \{x \le D, \quad x, & y = 2L + D, \quad y\}. \\
& \qquad \mathtt{t} \ , & & \\
& \quad \mathsf{STOP} \langle \mathtt{data} \rangle & \{x \le D, \quad x, & z \ge 3L + W + D, \quad \emptyset\}. \\
& \qquad \mathtt{M} \to \mathtt{W} : \mathsf{STOP} & \{x \le W, \quad x, & y = 2L + D, \quad \emptyset\}. \\
& \qquad \mathtt{end} & & \\
& \quad \} & &
\end{array}
$$

Separated semantics

- Actions (send, receive, “storage into queues”) take no time
- Time is modelled separately as *time actions*

$$
\begin{array}{lll}
\mu\mathtt{t}. & \mathtt{M} \to \mathtt{W} : \langle \mathtt{task} \rangle & \{x = 0,\ \emptyset,\ L \leq y \leq L+1,\ y\}. \\
& \mathtt{W} \to \mathtt{M} : \langle \mathtt{data} \rangle & \{y \leq W,\ \emptyset,\ x = 2L + W,\ x\}. \\
& \mathtt{M} \to \mathtt{A} : \{\text{MORE}\langle \mathtt{data} \rangle & \{x \leq D,\ \emptyset,\ z \geq 3L + W + D,\ z\}. \\
& \quad\quad \mathtt{M} \to \mathtt{W} : \text{MORE}\langle \mathtt{task} \rangle & \{x \leq D,\ x,\ y = 2L + D,\ y\}. \\
& \quad\quad \mathtt{t}\ , & \\
& \quad \text{STOP}\langle \mathtt{data} \rangle & \{x \leq D,\ x,\ z \geq 3L + W + D,\ \emptyset\}. \\
& \quad\quad \mathtt{M} \to \mathtt{W} : \text{STOP} & \{x \leq W,\ x,\ y = 2L + D,\ \emptyset\}. \\
& \quad\quad \mathtt{end} & \\
& \quad \} &
\end{array}
$$

$$
\begin{bmatrix} x = 0 \\ y = 0 \\ z = 0 \end{bmatrix}
\xrightarrow{\mathtt{MW}!\,\mathit{Task}}
\begin{bmatrix} x = 0 \\ y = 0 \\ z = 0 \end{bmatrix}
\xrightarrow{L}
\begin{bmatrix} x = L \\ y = L \\ z = L \end{bmatrix}
\xrightarrow{\mathtt{MW}?\,\mathit{Task}}
\begin{bmatrix} x = L \\ y = 0 \\ z = L \end{bmatrix}
\longrightarrow \quad \ldots
$$

Specified semantics

$x = 0, \ y = 0$ | $\mathtt{p} \to \mathtt{q} : \langle \mathtt{int} \rangle \quad \{1 \leq x \leq 10, \quad \emptyset, \quad y > 10, \quad \emptyset\}.\ \mathtt{end}$

Specified semantics:

$\xrightarrow{\mathtt{pq}!\langle \mathtt{int} \rangle}$

$\xrightarrow{15}$

$\xrightarrow{1.99} \ \xrightarrow{\mathtt{pq}!\langle \mathtt{int} \rangle}$

Specified semantics

$x = 0,\ y = 0$

$\mathtt{p} \to \mathtt{q} : \langle \mathtt{int} \rangle \quad \{1 \leq x \leq 10,\ \ \emptyset,\ \ y > 10,\ \ \emptyset\}.\ \mathtt{end}$

Specified semantics:

Communication actions at times that violate the constraints are not allowed

✘ $\xrightarrow{\mathrm{pq}!\langle \mathtt{int} \rangle}$

$\xrightarrow{15}$

$\xrightarrow{1.99} \quad \xrightarrow{\mathrm{pq}!\langle \mathtt{int} \rangle}$

Specified semantics

$x = 0,\ y = 0 \quad \mathtt{p} \to \mathtt{q} : \langle \mathtt{int} \rangle \quad \{1 \le x \le 10,\ \ \emptyset,\ \ y > 10,\ \ \emptyset\}.\ \mathtt{end}$

Specified semantics:

✘ $\xrightarrow{\mathtt{pq}!\langle \mathtt{int} \rangle}$

Communication actions at times that violate the constraints are not allowed

✘ $\xrightarrow{15}$

Time actions that make constraints of some **ready** action unsatisfiable are not allowed

$\xrightarrow{1.99} \quad \xrightarrow{\mathtt{pq}!\langle \mathtt{int} \rangle}$

Rethinking realizability

- Is this a good choreography?

$$\mathtt{p} \to \mathtt{q} : \langle \mathtt{Int} \rangle \{x > 3,\ \emptyset,\ y = 4,\ \emptyset\}.\mathtt{end}$$

$$\begin{matrix} x_{\mathrm{p}} = 0 \\ x_{\mathrm{q}} = 0 \end{matrix} \xrightarrow{\quad 5 \quad} \begin{matrix} x_{\mathrm{p}} = 5 \\ x_{\mathrm{q}} = 5 \end{matrix} \xrightarrow{\mathrm{pq}!Int} \begin{matrix} x_{\mathrm{p}} = 5 \\ x_{\mathrm{q}} = 5 \end{matrix}$$

- No, constraints may become unsatisfiable for $\mathtt{q}$ at some point

Rethinking realizability

Feasibility : *"a process can find a valid forward path until it reaches the end"*

- Not feasible

$$\mathtt{p} \to \mathtt{q} : \langle \mathtt{Int} \rangle \{x > 3,\ \emptyset,\ y = 4,\ \emptyset\}.\mathtt{end}$$

- Some feasible alternatives…

$$\mathtt{p} \to \mathtt{q} : \langle \mathtt{Int} \rangle \{x > 3 \wedge x \leq 4,\ \emptyset,\ y = 4,\ \emptyset\}.\mathtt{end}$$

$$\mathtt{p} \to \mathtt{q} : \langle \mathtt{Int} \rangle \{x > 3,\ \emptyset,\ y \geq 4,\ \emptyset\}.\mathtt{end}$$

Rethinking realizability

- Is this a good choreography?

$$\mathtt{p} \to \mathtt{q} : \langle \mathtt{Int} \rangle \{ x_{\mathtt{p}} < 3 \lor x_{\mathtt{p}} > 3,\ \ x_{\mathtt{q}} < 3 \lor x_{\mathtt{q}} > 3 \}.\mathtt{G}$$

- No, it may lead to inconsistent implementations

$$P = \mathtt{delay}(6).\ s[\mathrm{q}]!\langle \mathrm{p}, 10 \rangle; \mathtt{G} \downarrow \mathtt{p}$$
$$Q = s[\mathrm{p}]?(\mathrm{q}, x).\mathtt{G} \downarrow \mathtt{q}$$

Wait-freedom: *The constraint of each receive action must not admit, as a solution, a time which is earlier than some solution of the corresponding send action.*

A (very) simple timed calculus

- Session pi-calculus +delay:

$$\mathtt{delay}(t).P$$

$?\mathtt{M}(\mathit{Task})\{400 \leq x_{\mathtt{M}} \leq 401\}.$
$!\mathtt{M}\langle \mathit{Data}\rangle\{x_{\mathtt{M}} \leq 300401\}.$
$\ldots$

$\mathtt{delay}(400).$
$s[\mathtt{W}]?(\mathtt{M}, x).$
$\mathtt{delay}(100000).$
$s[\mathtt{W}]!\langle \mathtt{M}, \mathtt{sample}(x)\rangle).$
$\ldots$

Typing (some ideas…)

typing environment

$$\Gamma \vdash P \triangleright \Delta$$

session environment

$$c : (\nu, T)$$

virtual clock

$$\frac{\Gamma \vdash P \triangleright \{c_i:(\nu_i + t, \mathtt{T}_i)\}_{i\in I}}{\Gamma \vdash \mathtt{delay}(t).P \triangleright \{c_i:(\nu_i, \mathtt{T}_i)\}_{i\in I}} \quad \text{[Delay]}$$

$$\frac{j\in I \quad \Gamma\vdash e:S_j \quad \nu\models\delta_j \quad \Gamma \vdash P \triangleright \Delta, c:([\lambda_j\mapsto 0]\nu, \mathtt{T}_j)}{\Gamma \vdash c[\mathtt{p}]\triangleleft l_j\langle e\rangle;P \triangleright \Delta, c:(\nu, \mathtt{p}\oplus\{l_i\langle S_i\rangle\{\delta_i,\lambda_i\}.\mathtt{T}_i\}_{i\in I})} \quad \text{[Select]}$$

Properties

- Verification of real-time interactions with Multiparty Session Types
  - ***time-error freedom***: interactions are punctual
  - ***time-progress***: each reachable states is not a deadlock state and time can diverge (no Zeno)

Theorem (Time-error freedom) If $\Gamma \vdash P \triangleright \Delta$, and $P \longrightarrow^* P'$ then $P' \neq \texttt{error}$

Theorem (Progress for interleaved sessions) Let P be *feasible* and *wait-free*, $\Gamma \vdash P_0 \triangleright \emptyset$, $P_0 \rightarrow^+ P$. If P_0 is *session delay*, $\texttt{erase}(P)$ is not a deadlock process and if $\texttt{erase}(P) \rightarrow$ then $P \rightarrow$.

✘ $\texttt{delay}(6).\ \overline{a}[1].P \mid a[1].Q$

✔ $\overline{a}[1].\texttt{delay}(6).P \mid a[1].Q$

Timed MPST & CTA

Theorem (Soundness and completeness) (1) Let G be projectable then $\mathcal{A}(G)$ is basic and multiparty compatible. Furthemore with a specified semantics $G \approx \mathcal{A}(G)$. (2) If $\mathcal{C}$ is basic, multiparty compatible and with a specified semantics then there exists G such that $\mathcal{C} \approx \mathcal{A}(G)$.

Bocchi,Wang,Yoshida@CONCUR'14
"*Timed Multiparty Session Types*"

Timed MPST & CTA

Theorem (Progress and liveness of CTA) If $\mathcal{C}$ is basic, multiparty compatible and with a specified semantics and there exists a *feasible* G s.t. $\mathcal{C} \approx \mathcal{A}(G)$, then $\mathcal{C}$ satisfies progress and liveness.

- Progress: each reachable state is non-deadlock and allows time divergence
- Liveness: a final state can be reached from all reachable states

Bocchi,Wang,Yoshida@CONCUR'14
"*Timed Multiparty Session Types*"

- More permissive notion of feasibility, synthesis, more properties

Bocchi,Lange,Yoshida@CONCUR'15
"*Meeting Deadlines Together*"

Wrapping up

- Practical enough for a prototype

> Neykova,Bocchi,Yoshida@BEAT'14
> "*Timed Runtime Monitoring for Multiparty Conversations*"

- Several aspects of current work uncritical for wide use
  - Separated semantics
  - Semantics closer to automata models than real world
  - Very prescriptive timing in choreographies
  - Inflexible (strictly static) definition of delays in programs

Future work

- Time-sensitive protocol design and implementation EPSRC - EP/N035372/1
  - expressiveness
    - set flexible schedules for the timing of actions
    - support run-time adjustments
  - tractability
  - practicality

http://gow.epsrc.ac.uk/NGBOViewGrant.aspx?GrantRef=EP/N035372/1

Agenda

- MPSTs extensions
- Design by Contract
- Beyond static typing
- Relationship with automata
- Time
- Realizability

Back to choreographies

- Are all choreographies realisable as the composition of concurrent processes ?

- Choreographies and realisations are sets of traces

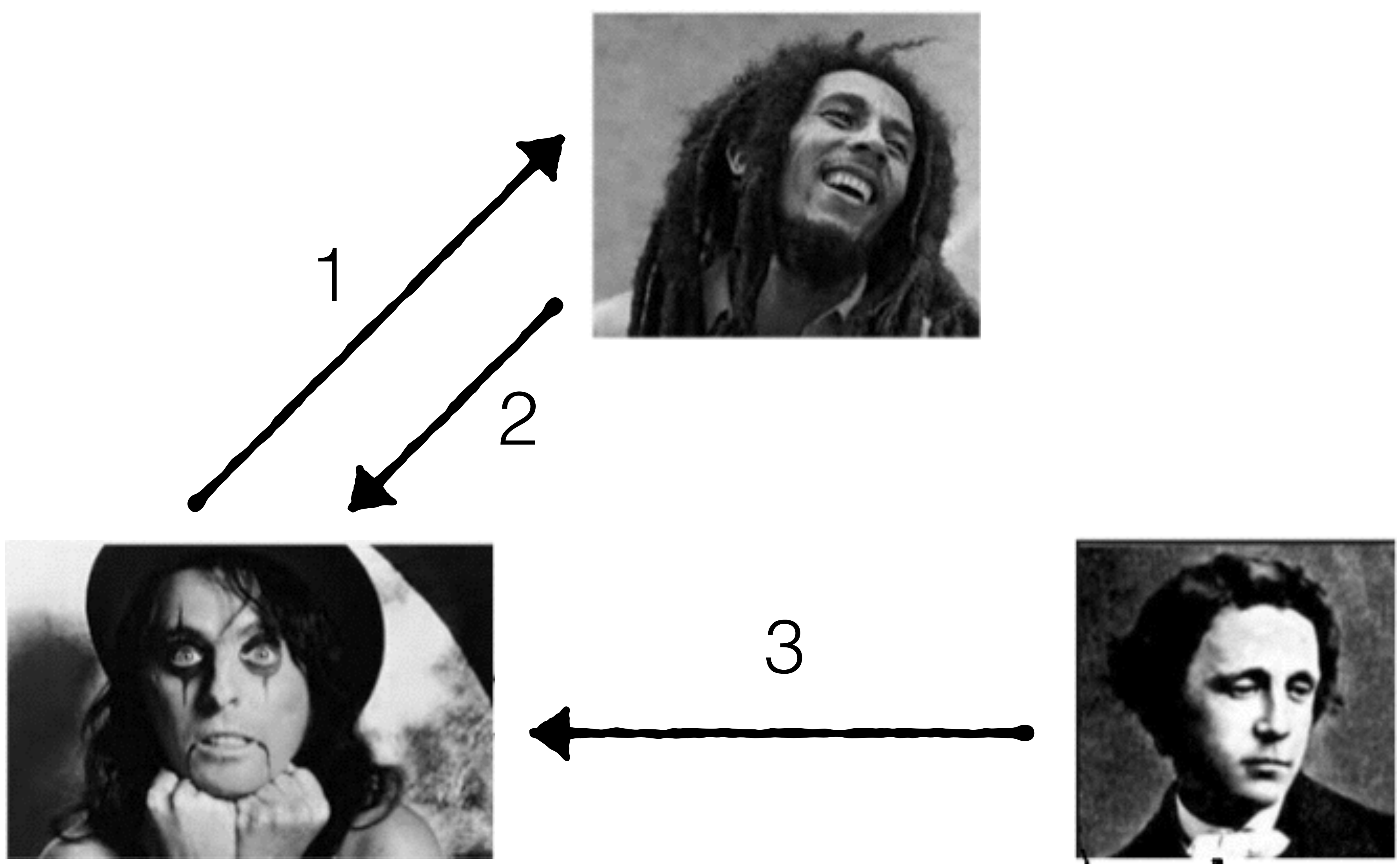
$$\ell \quad ::= \quad \mathtt{pq}!U \quad | \quad \mathtt{pq}?U$$

- Correct realisation (for now)

$$Traces(R) \subseteq Traces(C)$$

Realizability

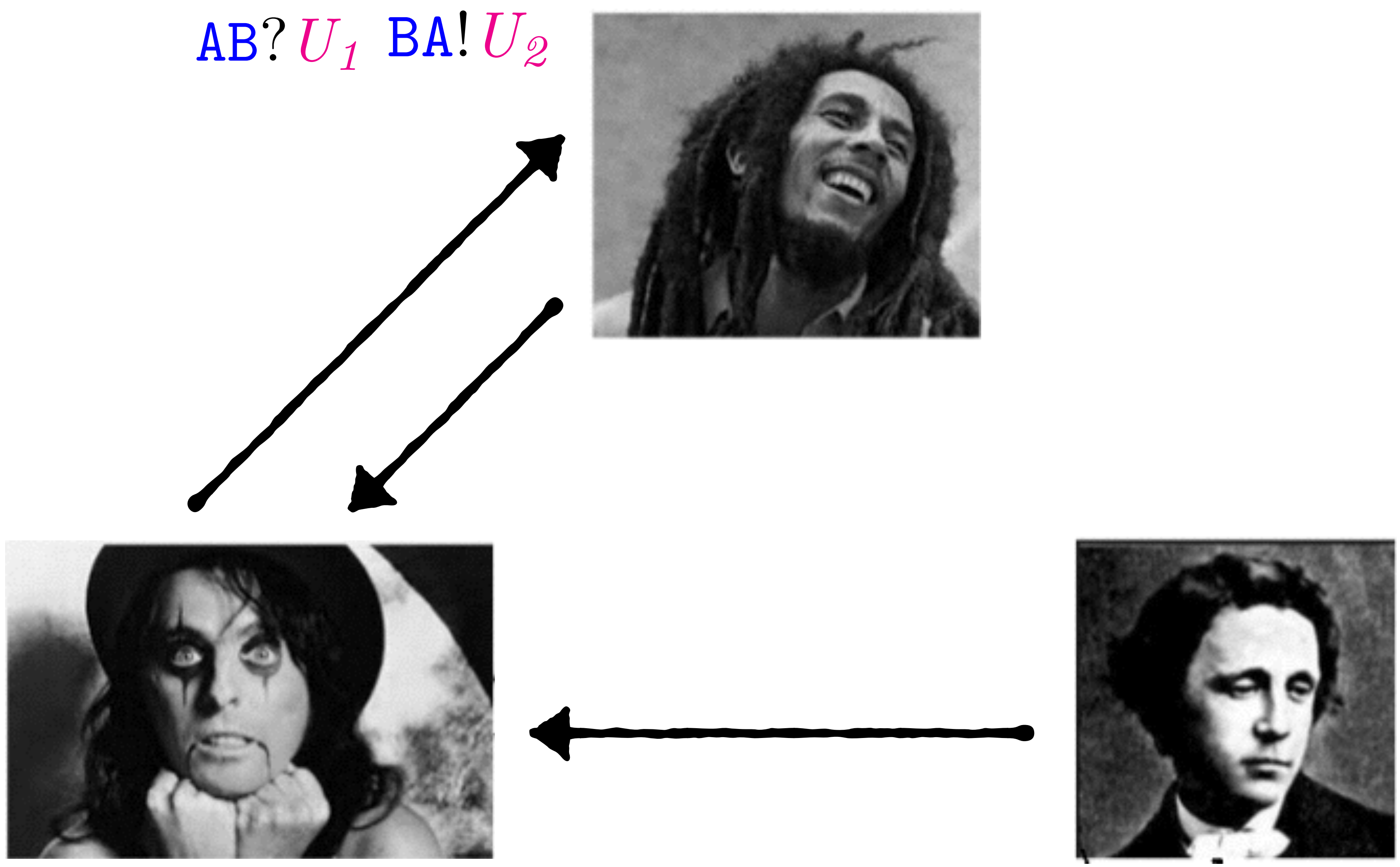
- What we want : $\mathtt{AB}!U_1$, $\mathtt{AB}?U_1$, $\mathtt{BA}!U_2$, $\mathtt{BA}?U_2$, $\mathtt{CA}!U_3$, $\mathtt{CA}?U_3$



Realizability

- What we want : $\mathtt{AB}!U_1$, $\mathtt{AB}?U_1$, $\mathtt{BA}!U_2$, $\mathtt{BA}?U_2$, $\mathtt{CA}!U_3$, $\mathtt{CA}?U_3$

$\mathtt{AB}?U_1$ $\mathtt{BA}!U_2$



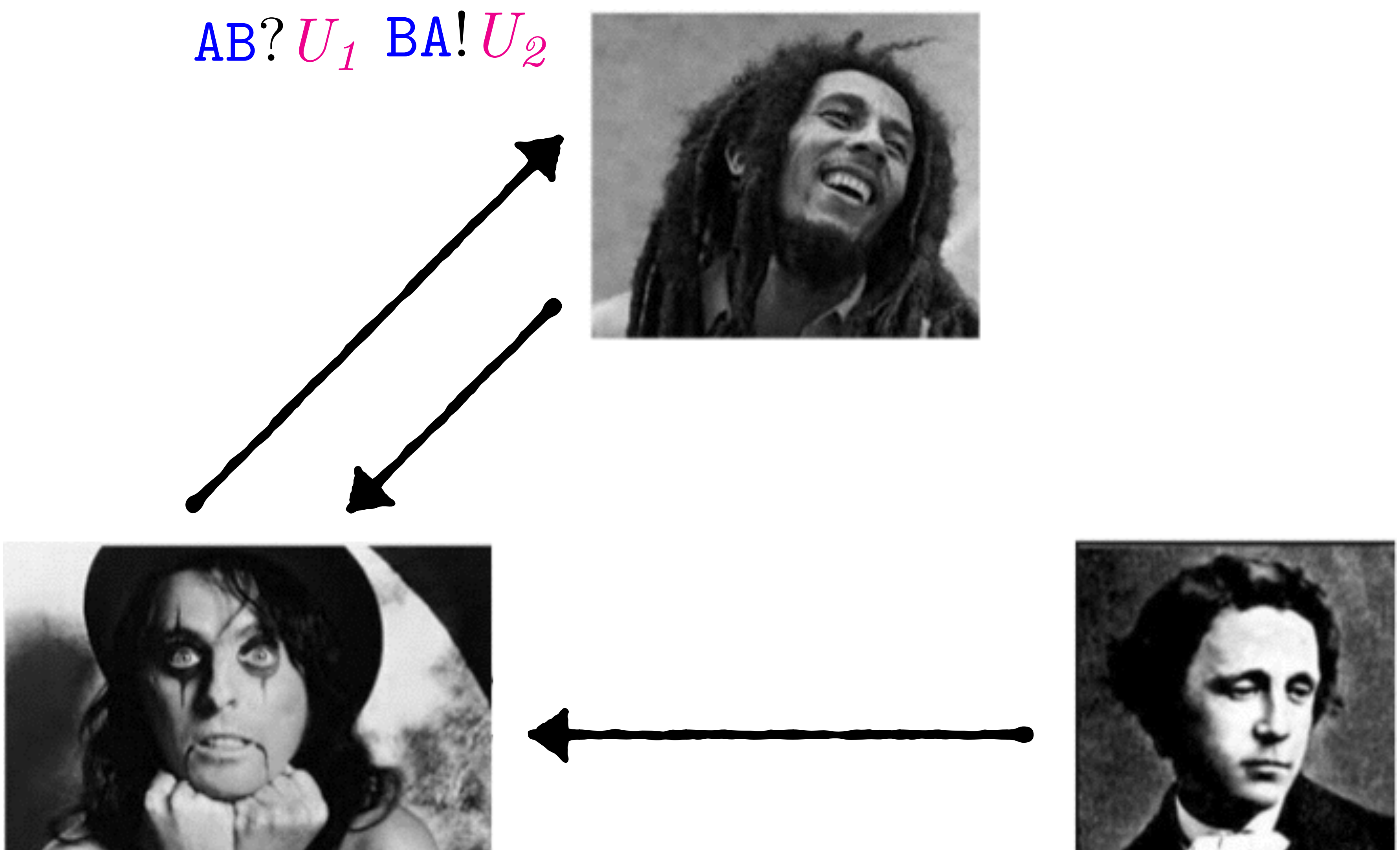
$\mathtt{AB}!U_1$ $\mathtt{BA}?U_2$ $\mathtt{CA}?U_3$

$\mathtt{CA}!U_3$

Realizability

- What we want : $\mathtt{AB}!U_1$, $\mathtt{AB}?U_1$, $\mathtt{BA}!U_2$, $\mathtt{BA}?U_2$, $\mathtt{CA}!U_3$, $\mathtt{CA}?U_3$
- What we may get : $\mathtt{CA}!U_3$, $\mathtt{AB}!U_1$, $\mathtt{AB}?U_1$, $\mathtt{BA}!U_2$, $\mathtt{BA}?U_2$, $\mathtt{CA}?U_3$

$\mathtt{AB}?U_1$ $\mathtt{BA}!U_2$



$\mathtt{AB}!U_1$ $\mathtt{BA}?U_2$ $\mathtt{CA}?U_3$

$\mathtt{CA}!U_3$

Realizability with MPST

- You cannot model the choreography that only allows trace

$$\mathtt{AB}!\,U_1,\ \mathtt{AB}?\,U_1,\ \mathtt{BA}!\,U_2,\ \mathtt{BA}?\,U_2,\ \mathtt{CA}!\,U_3,\ \mathtt{CA}?\,U_3$$

  using global types

- E.g., $\mathtt{A} \to \mathtt{B} : \langle U_1 \rangle .\ \mathtt{B} \to \mathtt{A} : \langle U_2 \rangle .\ \mathtt{C} \to \mathtt{A} : \langle U_3 \rangle .\ \mathtt{end}$

  also allows $\mathtt{CA}!\,U_3,\ \mathtt{AB}!\,U_1,\ \mathtt{AB}?\,U_1,\ \mathtt{BA}!\,U_2,\ \mathtt{BA}?\,U_2,\ \mathtt{CA}?\,U_3$

Realizability with MPST

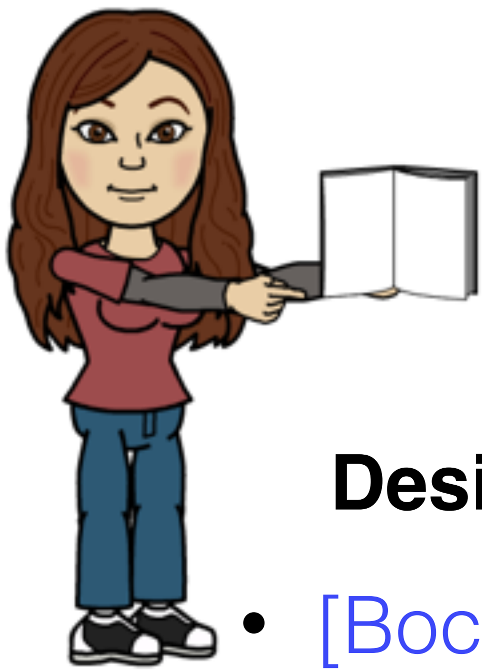
- Another example, that we have already seen is:

$$
\begin{array}{lll}
G & ::= & \mathtt{M} \to \mathtt{W} : \langle task \rangle . \\
 & & \mathtt{W} \to \mathtt{A} \{ \ \mathsf{ok} : \mathtt{W} \to \mathtt{A} : \langle data \rangle . \ \mathtt{A} \to \mathtt{M} : \langle result \rangle . \mathtt{end}, \\
 & & \qquad \mathsf{stop} : \ \mathtt{A} \to \mathtt{M} : \langle error\_code \rangle . \mathtt{end} \ \}
\end{array}
$$

which is not projectable (on $\mathtt{M}$)

- Projectable global types model realizable choreographies:

$$
\begin{tikzcd}
\textbf{Projectable} \arrow[r, bend right, "\text{yes}"'] & \textbf{Realizable} \arrow[l, bend right, "\text{it would be nice!}"']
\end{tikzcd}
$$



References

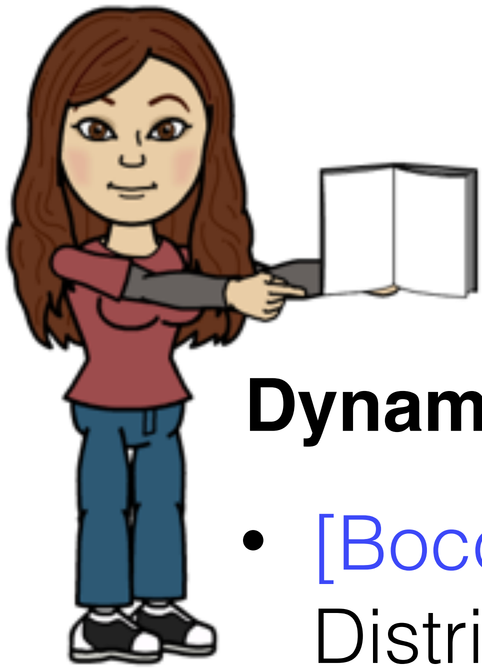
Design by Contract

- [Bocchi et al.@CONCUR'10] A theory of Design by Contract for Distributed Multiparty Interactions
- [Bocchi,Demangeon,Yoshida@TGC'13] A Multiparty Multi-Session Logic. *(extension with persistent variables)*
- [Bocchi,Demangeon@PLACES'13] Embedding Session Types in HML

Multiparty + Time

- [Bocchi,Yang,Yoshida@CONCUR'14] Timed Multiparty Session Types
- [Neykova,Bocchi,Yoshida@BEAT'14] Timed Runtime Monitoring for Multiparty Conversations
- [Bocchi,Lange,Yoshida@CONCUR'15] Meeting Deadlines Together

References

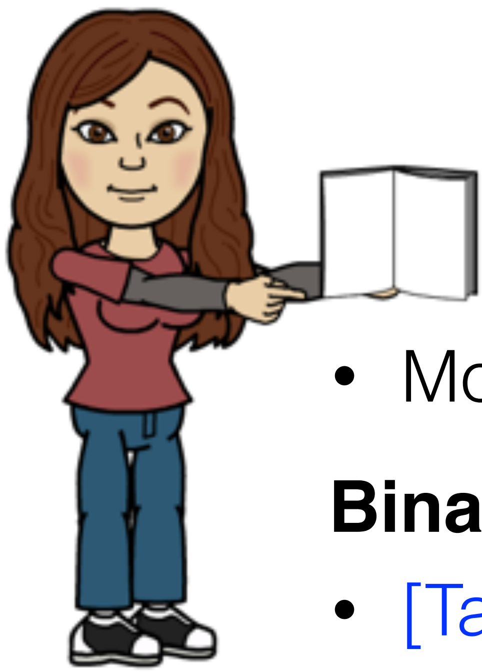


Dynamic Monitoring

- [Bocchi et al.@FORTE'13] A theory of Design by Contract for Distributed Multiparty Interactions
- [Chen et al.@TGC'13] Monitoring Networks through Multiparty Session Types.

Automata

- [Deniélou,Yoshida@ESOP'12] Multiparty Session Types Meet Communicating Automata
- [Deniélou,Yoshida@ICALP'13] Multiparty Compatibility in Communicating Automata: Characterisation and Synthesis of Global Session Types
- [Lange,Tuosto,Yoshida@POPL'15] From communicating machines to graphical choreographies
- [Bocchi,Lange,Yoshida@CONCUR'15] Meeting Deadlines Together



References

- Mobility Reading Group's home page **http://mrg.doc.ic.ac.uk**

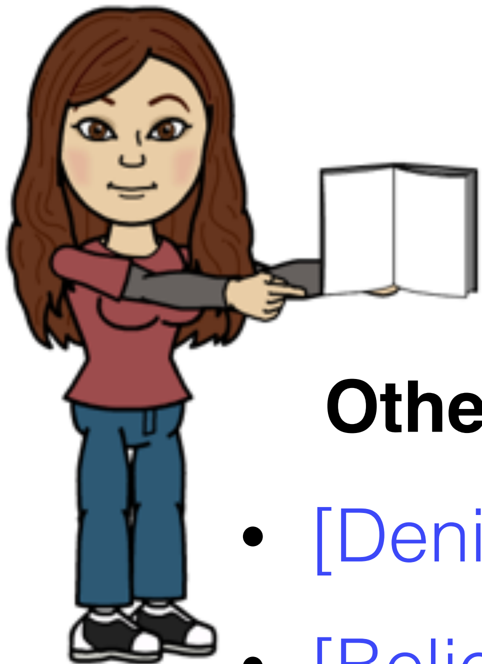
Binary

- [Takeuchi,Honda,Kubo@PARLE'94] An Interaction-Based Language and its Typing System
- [Honda,Vasconcelos,Kubo@ESOP'98] Language Primitives and Type Disciplines for Structured Communication-based Programming

Multiparty

- [Honda,Yoshida,Carbone@POPL'08] Multiparty asynchronous session types
- [Bettini et al.@CONCUR'08] Global Progress in Dynamically Interleaved Multiparty Sessions
- [Castagna, Dezani-Ciancaglini, Padovani@FTDS'12] On Global Types and Multi-Party Sessions
- [Deniélou,Yoshida@POPL'11] Dynamic multirole session types
- [Coppo et al.@SFM'15] Gentle Introduction to Multiparty Asynchronous Session Types

References



Others

- [Deniélou,Yoshida@POPL'11] Dynamic Multirole Session Types.
- [Beljeri,Yoshida@PLACES'08] Synchronous Multiparty Session types
- [Kouzapas,Yoshida@CONCUR'13] Globally Governed Session Semantics
- [Honda et al@COB'12] Structuring Communication with Session Types
- [Bocchi,Melgratti,Tuosto@ESOP'15] Resolving Non-Determinism in Choreographies

Scribble

- [Honda et al@COB'12] Structuring Communication with Session Types
- [Yoshida et al.@TGC'13] The Scribble protocol language