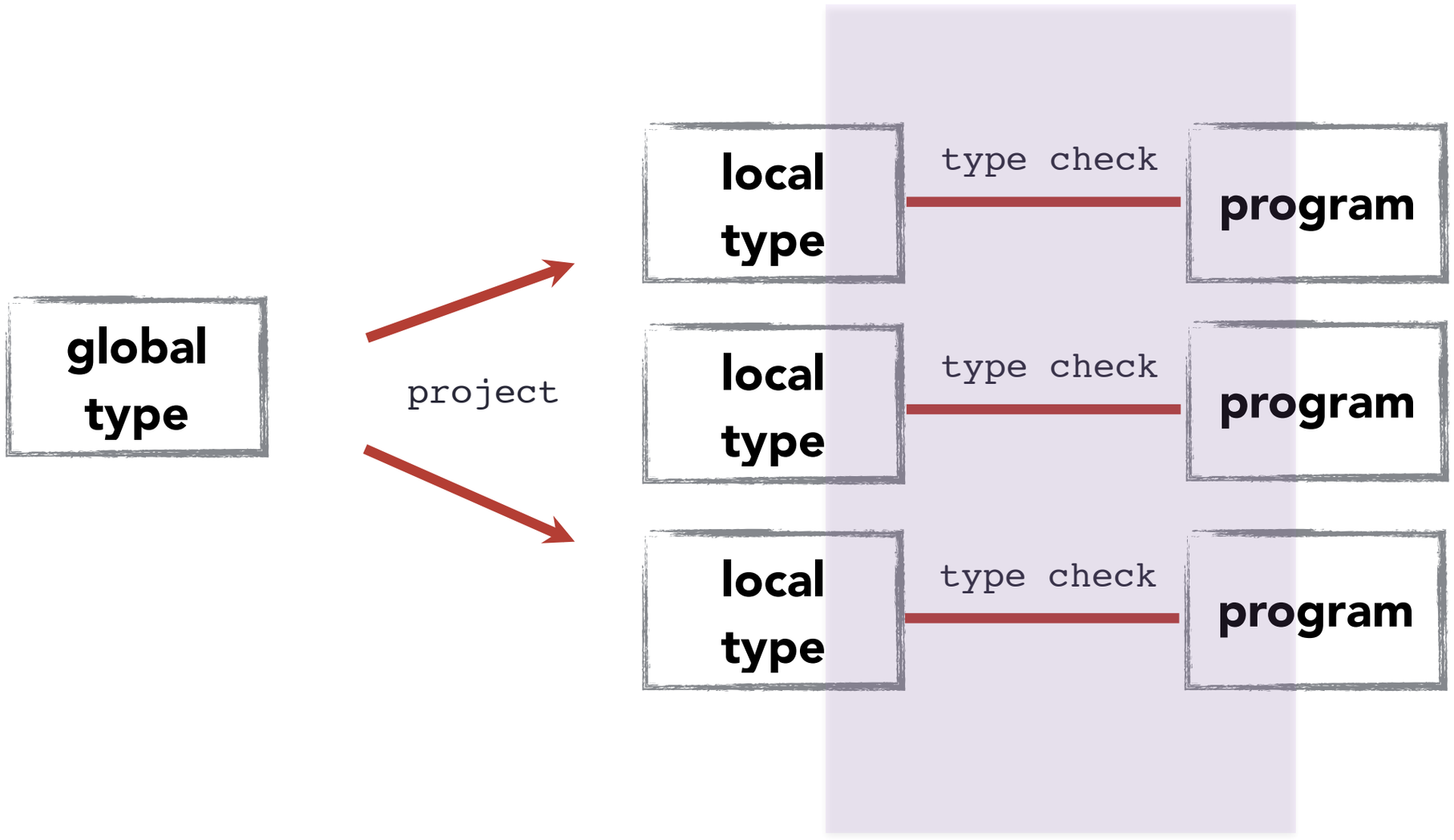
Multiparty Session Types

BETTY Summer School 2016

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

Today, typing



Agenda

- Processes vs types : an intuition
- What can go wrong?
- Typing
- Sending session types

Processes vs types

$$
\begin{array}{lll}
(\nu s) & & s[\mathtt{A}]!\langle \mathtt{B}, 10\rangle.s[\mathtt{A}]\&(\mathtt{C}, \{\mathsf{l}_i : P_i\}_{i\in\{1,2\}}) \\
& \mid & s[\mathtt{B}]?(\mathtt{A}, x).s[\mathtt{B}]!\langle \mathtt{C}, x\rangle.Q \\
& \mid & s[\mathtt{C}]?(\mathtt{B}, y).\mathtt{if}\ (y < 50)\ \mathtt{then}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathsf{l}_1\rangle.R_1 \\
& & \qquad\qquad\qquad\qquad\qquad\qquad \mathtt{else}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathsf{l}_2\rangle.R_2 \\
& \mid & s : \emptyset
\end{array}
$$

Processes vs types

$$!\langle \mathtt{B}, \mathit{Int}\rangle.\ \&\langle \mathtt{C}, \{\mathtt{l}_i : T_i\}_{i\in\{1,2\}}\rangle$$

$$\begin{aligned}
(\nu s)\quad & s[\mathtt{A}]!\langle \mathtt{B}, 10\rangle.s[\mathtt{A}]\&(\mathtt{C}, \{\mathtt{l}_i : P_i\}_{i\in\{1,2\}}) \\
\mid\quad & s[\mathtt{B}]?(\mathtt{A}, x).s[\mathtt{B}]!\langle \mathtt{C}, x\rangle.Q \\
\mid\quad & s[\mathtt{C}]?(\mathtt{B}, y).\mathtt{if}\ (y < 50)\ \mathtt{then}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathtt{l}_1\rangle.R_1 \\
& \qquad\qquad\qquad\qquad\qquad\qquad \mathtt{else}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathtt{l}_2\rangle.R_2 \\
\mid\quad & s : \emptyset
\end{aligned}$$

- send actual value (hopefully well-typed)
- branching is very similar

Processes vs types

$?(\mathtt{A}, Int).\ !\langle \mathtt{C}, Int \rangle.T_Q$

$$
\begin{array}{lll}
(\nu s) & & s[\mathtt{A}]!\langle \mathtt{B}, 10\rangle.s[\mathtt{A}]\&(\mathtt{C}, \{\mathtt{l}_i : P_i\}_{i\in\{1,2\}}) \\
& \mid & s[\mathtt{B}]?(\mathtt{A}, x).s[\mathtt{B}]!\langle \mathtt{C}, x\rangle.Q \\
& \mid & s[\mathtt{C}]?(\mathtt{B}, y).\mathtt{if}\ (y < 50)\ \mathtt{then}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathtt{l}_1\rangle.R_1 \\
& & \qquad\qquad\qquad\qquad\ \mathtt{else}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathtt{l}_2\rangle.R_2 \\
& \mid & s : \emptyset
\end{array}
$$

- send actual value (hopefully well-typed)
- branching is very similar
- receive actual value (instantiated in the continuation)
- let sent values depend on previous interactions

Processes vs types

$$?\langle \mathtt{B}, \mathit{Int}\rangle.\ \oplus\langle \mathtt{A}, \{\mathsf{l}_i : T'_i\}_{i\in\{1,2\}}\rangle$$

$$\begin{array}{lll}
(\nu s) & & s[\mathtt{A}]!\langle \mathtt{B}, 10\rangle.s[\mathtt{A}]\&(\mathtt{C}, \{\mathsf{l}_i : P_i\}_{i\in\{1,2\}}) \\
& | & s[\mathtt{B}]?(\mathtt{A}, x).s[\mathtt{B}]!\langle \mathtt{C}, x\rangle.Q \\
& | & s[\mathtt{C}]?(\mathtt{B}, y).\mathtt{if}\ (y < 50)\ \mathtt{then}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathsf{l}_1\rangle.R_1 \\
& & \qquad\qquad\qquad\qquad \mathtt{else}\ s[\mathtt{C}] \oplus \langle \mathtt{A}, \mathsf{l}_2\rangle.R_2 \\
& | & s : \emptyset
\end{array}$$

- send actual value (hopefully well-typed)
- branching is very similar
- receive actual value (instantiated in the continuation)
- let sent values depend on previous interactions
- how about select?

Wrapping up: processes vs types

- specification vs execution
- programs can implement many sessions

.....

- types vs actual values
- non-determinism vs conditional

.....

- delegation (will see…)

Agenda

- Processes vs types : an intuition
- What can go wrong?
- Typing
- Sending session types

What can go wrong?

communication safety

communication mismatch

session fidelity

unexpected behaviour

progress

deadlock

Communication safety

- Type errors
  - the exchanged data does not have the expected type

$$s[\mathtt{p}]?(\mathtt{q}, x).\mathbf{0} \mid s[\mathtt{q}]!\langle \mathtt{p}, \text{“Apple”}\rangle.\mathbf{0}$$

  - wrong values applies to an operator (e.g., in an expression)

$$s[\mathtt{p}]?(\mathtt{q}, x).s'[\mathtt{r}]!\langle \mathtt{s}, x+1\rangle.\mathbf{0} \mid s[\mathtt{q}]!\langle \mathtt{p}, \text{“Apple”}\rangle.\mathbf{0}$$

  - label mismatch

$$s[\mathtt{p}]\&(\mathtt{q}, \{\mathsf{l}_i : \mathbf{0}\}_{i\in\{1,2,3\}}) \mid s[\mathtt{q}] \oplus \langle \mathtt{p}, \mathsf{l}_4 : \mathbf{0}\rangle$$

Communication safety

- Linearity errors

$$s[\mathtt{p}]?(\mathtt{q}, y).P \mid s[\mathtt{p}]?(\mathtt{q}, y').P'$$

$$s[\mathtt{p}]!\langle \mathtt{q}, v \rangle.P \mid s[\mathtt{p}]!\langle \mathtt{q}, w \rangle.P'$$

- Would this be bad?

$$s[\mathtt{p}]!\langle \mathtt{q}, v \rangle.P \mid s : (\mathtt{p}, \mathtt{q}, w)$$

Progress

- Processes that get stuck

$$s[\mathtt{A}]?(\mathtt{B}, x).s[\mathtt{A}]!\langle \mathtt{B}, 5\rangle.\mathbf{0} \qquad a[\mathtt{A}](y).\overline{a}[\mathtt{A}](y').y!\langle \mathtt{B}, 5\rangle.y?(\mathtt{B}, x).\mathbf{0}$$

$$s[\mathtt{B}]?(\mathtt{A}, x).s[\mathtt{B}]!\langle \mathtt{A}, x\rangle.\mathbf{0} \qquad \overline{a}[\mathtt{B}](y).a[\mathtt{B}](y').y?(\mathtt{A}, x).y!\langle \mathtt{A}, x\rangle.\mathbf{0}$$

Typing rules

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

typing environment

service names u : <G>
variables x : S and X : S T

session environment

“ongoing” sessions c : T

Typing rules

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

typing environment

  service names u : <G>
  variables x : S and X : S T

session environment

  “ongoing” sessions c : T

[Macc] $$\frac{\Gamma \vdash u : \langle G \rangle \quad \Gamma \vdash P \triangleright \Delta, y : G \restriction \mathtt{p}}{\Gamma \vdash u[\mathtt{p}](y).P \triangleright \Delta}$$

Typing rules

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

typing environment

service names u : <G>
variables x : S and X : S T

session environment

“ongoing” sessions c : T

[Send]

$$\frac{\Gamma \vdash e : S \quad \Gamma \vdash P \triangleright \Delta, c : T}{\Gamma \vdash c!\langle \mathtt{p}, e\rangle.P \triangleright \Delta, c :!\langle \mathtt{p}, S\rangle.T}$$

[Rcv]

$$\frac{\Gamma, x : S \vdash P \triangleright \Delta, c : T}{\Gamma \vdash c?(\mathtt{p}, x).P \triangleright \Delta, c :?(\mathtt{p}, S).T}$$

Typing rules

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

typing environment

service names u : <G>
variables x : S and X : S T

session environment

“ongoing” sessions c : T

[Branch] $$\frac{\Gamma \vdash P_i \triangleright \Delta, c : T_i \qquad \forall i \in I}{\Gamma \vdash c\&(\mathtt{p}, \{\mathsf{l}_i : P_i\}_{i \in I}) \triangleright \Delta, c : \&(\mathtt{p}, \{\mathsf{l}_i : T_i\}_{i \in I})}$$

Example

$$
\begin{aligned}
G = \quad & \mathtt{A} \to \mathtt{S} : \langle \mathit{String} \rangle. \\
& \mathtt{S} \to \{\mathtt{A}, \mathtt{B}\} : \langle \mathit{Int} \rangle. \\
& \mathtt{A} \to \mathtt{B} : \langle \mathit{Int} \rangle. \\
& \mathtt{B} \to \{\mathtt{S}, \mathtt{A}\} : \{\mathsf{ok} : \mathtt{B} \to \mathtt{S} : \langle \mathit{String} \rangle. \\
& \qquad\qquad\qquad\qquad \mathtt{S} \to \mathtt{B} : \langle \mathit{Date} \rangle.\mathtt{end}, \\
& \qquad\qquad\qquad \mathsf{quit} : \mathtt{end}\}
\end{aligned}
$$

Example

$$G \upharpoonright \mathtt{A} = !\langle \mathtt{S}, \mathit{String} \rangle.\ ?(\mathtt{S}, \mathit{Int}).\ !\langle \mathtt{B}, \mathit{Int} \rangle.\ \&(\mathtt{B}, \{\mathsf{ok} : \mathtt{end}, \mathsf{quit} : \mathtt{end}\})$$

$$\begin{aligned} P_{\mathtt{A}} \quad = \quad & y!\langle \mathtt{S}, \text{“}\mathit{Notes\ from\ the\ Underground}\text{”} \rangle. \\ & y?(\mathtt{S}, x_{quote}). \\ & y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2 \rangle. \\ & y\&(\mathtt{B}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\}) \end{aligned}$$

$$\frac{\Gamma \vdash a : \langle G \rangle \qquad \Gamma \vdash P_{\mathtt{A}} \triangleright y : G \upharpoonright \mathtt{A} \qquad \mathtt{A} \in \mathcal{P}(G)}{\Gamma \vdash a[\mathtt{A}](y).P_{\mathtt{A}} \triangleright \emptyset}\ [\text{Macc}]$$

Example

$$G \upharpoonright \mathtt{A} = !\langle \mathtt{S}, \mathit{String} \rangle.\ ?(\mathtt{S}, \mathit{Int}).\ !\langle \mathtt{B}, \mathit{Int} \rangle.\ \&(\mathtt{B}, \{\mathsf{ok} : \mathtt{end}, \mathsf{quit} : \mathtt{end}\})$$

$$\begin{aligned} P_{\mathtt{A}} \quad = \quad & y!\langle \mathtt{S}, \text{“}\mathit{Notes\ from\ the\ Underground}\text{”} \rangle. \\ & y?(\mathtt{S}, x_{quote}). \\ & y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2 \rangle. \\ & y\&(\mathtt{B}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\}) \end{aligned}$$

$$\frac{\Gamma \vdash \text{“}\mathit{Notes\ from\ the\ Underground}\text{”} : \mathit{String} \qquad \Gamma \vdash y?(\mathtt{S}, x_{quote}).P''_{\mathtt{A}} \triangleright y :?(\mathtt{S}, \mathit{Int}).T''_{\mathtt{A}}}{\Gamma \vdash y!\langle \mathtt{S}, \text{“}\mathit{Notes\ from\ the\ Underground}\text{”} \rangle.P'_{\mathtt{A}} \triangleright y :!\langle \mathtt{S}, \mathit{String} \rangle.T'_{\mathtt{A}}}\ [\text{Send}]$$

Example

$$G \upharpoonright \mathtt{A} = !\langle \mathtt{S}, \mathit{String}\rangle.\ ?(\mathtt{S}, \mathit{Int}).\ !\langle \mathtt{B}, \mathit{Int}\rangle.\ \&(\mathtt{B}, \{\mathsf{ok} : \mathtt{end}, \mathsf{quit} : \mathtt{end}\})$$

$$\begin{aligned} P_{\mathtt{A}} \quad = \quad & y!\langle \mathtt{S}, \text{“}\mathit{Notes\ from\ the\ Underground}\text{”}\rangle. \\ & y?(\mathtt{S}, x_{quote}). \\ & y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2\rangle. \\ & y\&(\mathtt{B}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\}) \end{aligned}$$

$$\frac{\Gamma, x_{quote} : \mathit{Int} \vdash y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2\rangle.P'''_{\mathtt{A}} \triangleright y :!\langle \mathtt{B}, \mathit{Int}\rangle.T'''_{\mathtt{A}}}{\Gamma \vdash y?(\mathtt{S}, x_{quote}).P''_{\mathtt{A}} \triangleright y :?(\mathtt{S}, \mathit{Int}).T''_{\mathtt{A}}}\ [\text{Rcv}]$$

Example

$$G \upharpoonright \mathtt{A} = !\langle \mathtt{S}, \mathit{String}\rangle.\ ?(\mathtt{S}, \mathit{Int}).\ !\langle \mathtt{B}, \mathit{Int}\rangle.\ \&(\mathtt{B}, \{\mathsf{ok} : \mathtt{end}, \mathsf{quit} : \mathtt{end}\})$$

$$\begin{aligned} P_{\mathtt{A}} \quad = \quad & y!\langle \mathtt{S}, \text{“}\mathit{Notes\ from\ the\ Underground}\text{”}\rangle. \\ & y?(\mathtt{S}, x_{quote}). \\ & y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2\rangle. \\ & y\&(\mathtt{B}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\}) \end{aligned}$$

$$\frac{\begin{array}{c}\Gamma, x_{quote} : \mathit{Int} \vdash x_{quote}\ \mathtt{div}\ 2 : \mathit{Int} \\ \Gamma, x_{quote} : \mathit{Int} \vdash P'''_{\mathtt{A}} \triangleright y : T'''_{\mathtt{A}}\end{array}}{\Gamma, x_{quote} : \mathit{Int} \vdash y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2\rangle.P'''_{\mathtt{A}} \triangleright y :!\langle \mathtt{B}, \mathit{Int}\rangle.T'''_{\mathtt{A}}}\ [\text{Send}]$$

Example

$$G \upharpoonright \mathtt{A} = !\langle \mathtt{S}, \mathit{String} \rangle.\ ?(\mathtt{S}, \mathit{Int}).\ !\langle \mathtt{B}, \mathit{Int} \rangle.\ \&(\mathtt{B}, \{\mathsf{ok} : \mathtt{end}, \mathsf{quit} : \mathtt{end}\})$$

$$\begin{aligned} P_{\mathtt{A}} \quad = \quad & y!\langle \mathtt{S}, \text{“}\mathit{Notes\ from\ the\ Underground}\text{”} \rangle. \\ & y?(\mathtt{S}, x_{quote}). \\ & y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2 \rangle. \\ & y\&(\mathtt{B}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\}) \end{aligned}$$

$$\dfrac{\dfrac{-}{\Gamma, x_{quote} : \mathit{Int} \vdash \mathbf{0} \triangleright y : \mathtt{end}} \qquad \dfrac{-}{\Gamma, x_{quote} : \mathit{Int} \vdash \mathbf{0} \triangleright y : \mathtt{end}}\ [\text{Inact}]}{\Gamma, x_{quote} : \mathit{Int} \vdash y\&(\mathtt{B}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\} \triangleright y : y\&(\mathtt{B}, \{\mathsf{ok} : \mathtt{end}, \mathsf{quit} : \mathtt{end}\})}\ [\text{Branch}]$$

The whole tree

$$\dfrac{\dfrac{-}{\Gamma, x_{quote} : \mathit{Int} \vdash \mathbf{0} \triangleright y : \mathtt{end}}\ [\text{Inact}] \qquad \dfrac{-}{\Gamma, x_{quote} : \mathit{Int} \vdash \mathbf{0} \triangleright y : \mathtt{end}}\ [\text{Inact}]}{\Gamma, x_{quote} : \mathit{Int} \vdash y\&(\mathsf{B}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\} \triangleright y : y\&(\mathsf{B}, \{\mathsf{ok} : \mathtt{end}, \mathsf{quit} : \mathtt{end}\})}\ [\text{Branch}]$$

$$\dfrac{\Gamma, x_{quote} : \mathit{Int} \vdash y!\langle \mathsf{B}, x_{quote}\ \mathtt{div}\ 2\rangle.P'''_{\mathtt{A}} \triangleright y :!\langle \mathsf{B}, \mathit{Int}\rangle.T'''_{\mathtt{A}}}{\Gamma, x_{quote} : \mathit{Int} \vdash y!\langle \mathsf{B}, x_{quote}\ \mathtt{div}\ 2\rangle.P'''_{\mathtt{A}} \triangleright y :!\langle \mathsf{B}, \mathit{Int}\rangle.T'''_{\mathtt{A}}}\ [\text{Send}]$$

$$\dfrac{\dfrac{\Gamma \vdash \textit{“Notes from the Underground”} : \mathit{String} \qquad \Gamma \vdash y?(\mathsf{S}, x_{quote}).P''_{\mathtt{A}} \triangleright y :?(\mathsf{S}, \mathit{Int}).T''_{\mathtt{A}}}{\Gamma \vdash y!\langle \mathsf{S}, \textit{“Notes from the Underground”}\rangle.P'_{\mathtt{A}} \triangleright y :!\langle \mathsf{S}, \mathit{String}\rangle.T'_{\mathtt{A}}}\ [\text{Rcv}]}{\Gamma \vdash y!\langle \mathsf{S}, \textit{“Notes from the Underground”}\rangle.P'_{\mathtt{A}} \triangleright y :!\langle \mathsf{S}, \mathit{String}\rangle.T'_{\mathtt{A}}}\ [\text{Send}]$$

Typing rules

typing environment

session environment

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

[Sel]
$$\frac{\Gamma \vdash P \triangleright \Delta, c : T_j \qquad j \in I}{\Gamma \vdash c \oplus \langle \mathtt{p}, \mathsf{l}_j \rangle . P \triangleright \Delta, c : \oplus(\mathtt{p}, \{\mathsf{l}_i : T_i\}_{i \in I})}$$

[Cond]
$$\frac{\Gamma \vdash e : \mathit{bool} \quad \Gamma \vdash P \triangleright \Delta \quad \Gamma \vdash Q \triangleright \Delta}{\Gamma \vdash \mathtt{if}\ e\ \mathtt{then}\ P\ \mathtt{else}\ Q \triangleright \Delta}$$

Example

$$\frac{\Gamma \vdash P_1 \triangleright y : T_1 \quad 1 \in \{1,2,3\}}{\Gamma \vdash y \oplus \langle \mathtt{p}, \mathtt{l}_1 : P_1 \rangle \triangleright y : \oplus \langle \mathtt{p}, \{\mathtt{l}_i : T_i\}_{i \in \{1,2,3\}} \rangle)} \text{ [Sel]}$$

$$\frac{\Gamma \vdash y \oplus \langle \mathtt{p}, \mathtt{l}_1 : P_1 \rangle \triangleright y : \oplus \langle \mathtt{p}, \{\mathtt{l}_i : T_i\}_{i \in \{1,2,3\}} \rangle) \quad \Gamma \vdash y \oplus \langle \mathtt{p}, \mathtt{l}_2 : P_2 \rangle \triangleright y : \oplus \langle \mathtt{p}, \{\mathtt{l}_i : T_i\}_{i \in \{1,2,3\}} \rangle)}{\Gamma \vdash \mathtt{if}\ x > 0\ \mathtt{then}\ y \oplus \langle \mathtt{p}, \mathtt{l}_1 : P_1 \rangle\ \mathtt{else}\ y \oplus \langle \mathtt{p}, \mathtt{l}_2 : P_1 \rangle \triangleright y : \oplus \langle \mathtt{p}, \{\mathtt{l}_i : T_i\}_{i \in \{1,2,3\}} \rangle)} \text{ [Cond]}$$

Typing rules

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

typing environment

service names u : <G>
variables x : S and X : S T

session environment

“ongoing” sessions c : T

[Conc]

$$\frac{\Gamma \vdash P \triangleright \Delta \quad \Gamma \vdash Q \triangleright \Delta' \quad \mathrm{dom}(\Delta) \cap \mathrm{dom}(\Delta') = \emptyset}{\Gamma \vdash P \mid Q \triangleright \Delta, \Delta'}$$

Typing rules

typing environment

session environment

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

[Def]

$$\frac{\Gamma, X : S\,\mathbf{t}, x : S \vdash P \triangleright y : T \qquad \Gamma, X : S\,\mu\mathbf{t}.T \vdash Q \triangleright \Delta}{\Gamma \vdash \mathsf{def}\ X(x, y) \ = \ P\ \mathsf{in}\ Q \triangleright \Delta}$$

[Var]

$$\frac{\Gamma \vdash e : S \qquad \Delta\ \mathtt{end}\ \text{only}}{\Gamma, X : S\,T \vdash X\langle e, c\rangle \triangleright \Delta, c : T}$$

Example (1/2)
derivation with recursive type

How do we derive a type of a recursive process with a boolean expression ?

def $X(y,x) = x!\langle 1,y\rangle.X\langle y,x\rangle$ in $X\langle \mathsf{true},c\rangle$

Let $T = !\langle 1, \mathsf{Bool}\rangle.\mathbf{t}$

$$\frac{\Gamma, X : S\,\mathbf{t}, x : S \vdash P \triangleright y : T \qquad \Gamma, X : S\,\mu\mathbf{t}.T \vdash Q \triangleright \Delta}{\Gamma \vdash \mathsf{def}\ X(x,y) = P\ \mathsf{in}\ Q \triangleright \Delta}$$

$$\dfrac{\Theta \qquad \dfrac{\vdash \mathsf{true} : \mathsf{Bool}}{X : \mathsf{Bool}\ \mu\mathbf{t}.T \vdash X\langle \mathsf{true},c\rangle \triangleright c : \mu\mathbf{t}.T}\,(\mathsf{Var})}{\vdash \mathsf{def}\ X(y,x) = x!\langle 1,y\rangle.X\langle y,x\rangle\ \mathsf{in}\ X\langle \mathsf{true},c\rangle \triangleright c : \mu\mathbf{t}.T}\,(\mathsf{Def})$$

from [Yoshida@SFM'14] (lecture slides)

Example (2/2) derivation with recursive type

$$\frac{\Gamma \vdash e : S \qquad \Delta \ \mathbf{end}\ \text{only}}{\Gamma, X : S\,T \vdash X\langle e, c\rangle \triangleright \Delta, c : T}$$

where Θ is the derivation :

$$\dfrac{y : \mathsf{Bool} \vdash y : \mathsf{Bool} \qquad \dfrac{y : \mathsf{Bool} \vdash y : \mathsf{Bool}}{X : \mathsf{Bool}\ \mathbf{t}, y : \mathsf{Bool} \vdash X\langle y, x\rangle \triangleright x : \mathbf{t}}\ (\mathsf{Var})}{X : \mathsf{Bool}\ \mathbf{t}, y : \mathsf{Bool} \vdash x!\langle 1, y\rangle.X\langle y, x\rangle \triangleright x :!\langle 1, \mathsf{Bool}\rangle.\mathbf{t}}\ (\mathsf{Send})$$

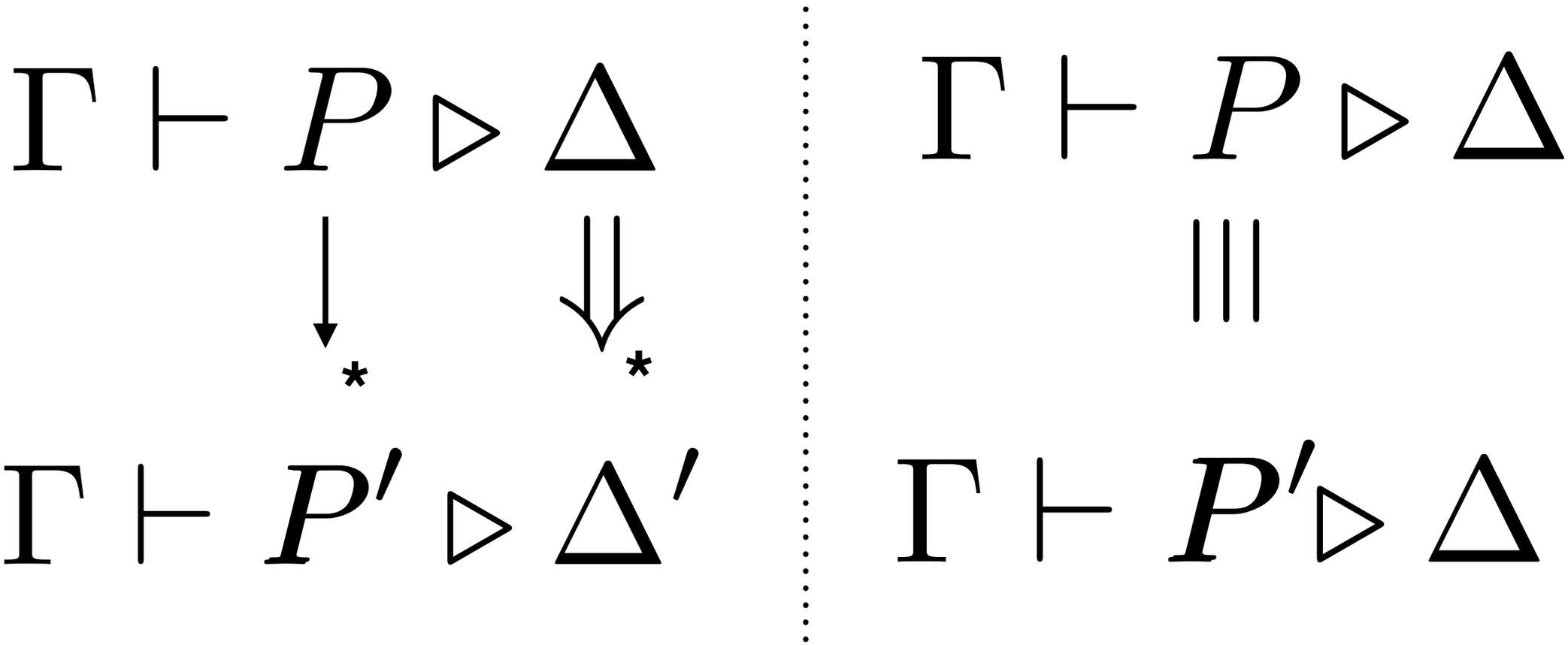
$$\frac{\Gamma, X : S\,\mathbf{t}, x : S \vdash P \triangleright y : T \qquad \Gamma, X : S\,\mu\mathbf{t}.T \vdash Q \triangleright \Delta}{\Gamma \vdash \mathsf{def}\ X(x, y) = P\ \mathsf{in}\ Q \triangleright \Delta}$$

from [Yoshida@SFM'14] (lecture slides)

The properties more formally

- **Subject Reduction**
  - session fidelity
    - communication safety
      - *error-freedom*
      - *linearity*
- **Progress**

Subject Reduction



Subject reduction

Subject congruence

Yields : **Communication Safety** & **Session fidelity**

Progress

$$a : G \vdash P_1 \mid \ldots \mid P_n \triangleright \emptyset$$

- each role of G is implemented by (exactly one) Pi
- no name hiding, no starting new sessions

$P_1 \mid \ldots \mid P_n \rightarrow^+ Q$ implies either

$Q \equiv \mathbf{0}$

$\exists R.Q \rightarrow R$

[Honda,Yoshida,Carbone@POPL'08]

.....

[Coppo et al.@CONCUR'08] **interaction typing system**
guarantees progress for interleaved sessions

Agenda

- Processes vs types : an intuition
- What can go wrong?
- Typing
- Sending session types

Sending session types

$$
\begin{array}{lll}
G & ::= & \mathtt{p} \to \mathtt{q} : \langle U \rangle.G \\
& \mid & \mathtt{p} \to \mathtt{q} : \{\mathsf{l}_i : G_i\}_{i \in I} \\
& \mid & \mu \mathtt{t}.\ G \\
& \mid & \mathtt{t} \\
& \mid & \mathtt{end}
\end{array}
$$

$$Int \mid \ldots \mid T \mid G$$

- e.g., $\mathtt{A} \to \mathtt{B} : \langle G \rangle.\mathtt{end}$
- What is a value of type ***G*** ?
    - A service (shared) channel!

Programs

$$
\begin{array}{lcl}
P & ::= & \overline{u}[\mathtt{p}](y).P \\
& | & u[\mathtt{p}](y).P \\
& | & c!\langle \mathtt{p}, e\rangle.P \\
& | & c?(\mathtt{p}, x).P \\
& | & c \oplus \langle \mathtt{p}, \mathsf{l}\rangle.P \\
& | & c\&(\mathtt{p}, \{\mathsf{l}_i : P_i\}_{i\in I}) \\
& | & c!\langle\langle \mathtt{p}, s'[\mathtt{p}']\rangle\rangle.P \\
& | & c?((\mathtt{p}, y)).P \\
& | & \text{if } e \text{ then } P \text{ else } Q \\
& | & P \mid Q \\
& | & \mathbf{0} \\
& | & (\nu a)P \\
& | & \text{def } D \text{ in } P \\
& | & X\langle e, c\rangle
\end{array}
$$

Example: sending type G

$$\texttt{A} \to \texttt{B} : \langle G \rangle.\texttt{end}$$

$$(\nu s)((\nu a)s[\texttt{A}]!\langle \texttt{B}, a\rangle.\overline{a}[\texttt{A}](y).P) \mid s[\texttt{B}]?(\texttt{A}, x).x[\texttt{B}](y).Q \mid s : \emptyset$$

Example: sending type G

$$(\nu s)((\nu a)s[\mathtt{A}]!\langle \mathtt{B}, a\rangle.\overline{a}[\mathtt{A}](y).P) \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : \emptyset$$

Example: sending type G

$$(\nu s)\ (\nu a)s[\mathtt{A}]!\langle \mathtt{B}, a\rangle.\overline{a}[\mathtt{A}](y).P \ \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : \emptyset$$

Example: sending type G

$$(\nu s)\ (\nu a)s[\mathtt{A}]!\langle \mathtt{B}, a\rangle.\overline{a}[\mathtt{A}](y).P \ \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : \emptyset$$

$$\rightarrow$$

$$(\nu s)\ (\nu a)\overline{a}[\mathtt{A}](y).P \ \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : (\mathtt{A}, \mathtt{B}, a)$$

Example: sending type G

$$(\nu s)\ (\nu a)s[\mathtt{A}]!\langle \mathtt{B}, a\rangle.\overline{a}[\mathtt{A}](y).P \ \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : \emptyset$$

$$\rightarrow$$

$$(\nu s)\ (\nu a)\overline{a}[\mathtt{A}](y).P \ \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : (\mathtt{A}, \mathtt{B}, a)$$

$$\rightarrow$$

$$(\nu s)(\nu a)\ \overline{a}[\mathtt{A}](y).P \mid a[\mathtt{B}](y).Q \mid s : \emptyset$$

Example: sending type G

$$(\nu s)\ (\nu a)s[\mathtt{A}]!\langle \mathtt{B}, a\rangle.\overline{a}[\mathtt{A}](y).P \ \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : \emptyset$$

$$\rightarrow$$

$$(\nu s)\ (\nu a)\overline{a}[\mathtt{A}](y).P \ \mid s[\mathtt{B}]?(\mathtt{A}, x).x[\mathtt{B}](y).Q \mid s : (\mathtt{A}, \mathtt{B}, a)$$

$$\rightarrow$$

$$(\nu s)(\nu a)\ \overline{a}[\mathtt{A}](y).P \mid a[\mathtt{B}](y).Q \mid s : \emptyset$$

$$\rightarrow$$

$$(\nu s)(\nu a)(\nu s')\ P[s'[\mathtt{A}]/y] \mid Q[s'[\mathtt{B}]/y] \mid s' : \emptyset \mid s : \emptyset$$

Sending session types

$$
\begin{array}{lll}
G & ::= & \mathtt{p} \to \mathtt{q} : \langle U \rangle . G \\
& \mid & \mathtt{p} \to \mathtt{q} : \{\mathsf{l}_i : G_i\}_{i \in I} \\
& \mid & \mu \mathtt{t}.\ G \\
& \mid & \mathtt{t} \\
& \mid & \mathtt{end}
\end{array}
$$

$$Int \mid \ldots \mid T \mid G$$

- If I need to send a value of type T…what should I send?
  - a session channel with role e.g, $s[\mathtt{p}]$
  - its type T is the remaining (delegated) behaviour

Delegation

$$
\begin{array}{lcll}
P & ::= & \overline{u}[\mathtt{p}](y).P & \\
 & | & u[\mathtt{p}](y).P & \\
 & | & c!\langle \mathtt{p}, e\rangle.P & \\
 & | & c?(\mathtt{p}, x).P & \\
 & | & c!\langle\!\langle \mathtt{p}, s'[\mathtt{p}']\rangle\!\rangle.P & \text{channel delegation} \\
 & | & c?((\mathtt{p}, y)).P & \text{channel reception} \\
 & | & \ldots &
\end{array}
$$

.....

[Deleg] $s[\mathtt{p}]!\langle\!\langle \mathtt{q}, s'[\mathtt{p}']\rangle\!\rangle.P \mid s : h \quad \rightarrow \quad P \mid s : h \cdot (\mathtt{p}, \mathtt{q}, s'[\mathtt{p}'])$

[SRcv] $s[\mathtt{p}]?((\mathtt{q}, y)).P \mid s : (\mathtt{p}, \mathtt{q}, s'[\mathtt{p}']) \cdot h \quad \rightarrow \quad P[y/s'[\mathtt{p}']] \mid s : h$

Example (delegation and types)

G1

$\mathtt{A} \to \mathtt{S} : \langle \mathit{String} \rangle.$
$\mathtt{S} \to \{\mathtt{A}, \mathtt{B}\} : \langle \mathit{Int} \rangle.$
$\mathtt{A} \to \mathtt{B} : \langle \mathit{Int} \rangle.$
$\mathtt{B} \to \mathtt{S} : \{\mathsf{ok} : \mathtt{B} \to \mathtt{S} : \langle \mathit{String} \rangle.$
$\mathtt{S} \to \mathtt{B} : \langle \mathit{Date} \rangle.\mathtt{end},$
$\mathsf{quit} : \mathtt{end}\}$

G2

$\mathtt{B} \to \mathtt{C} : \langle \mathit{String} \rangle.$
$\mathtt{B} \to \mathtt{C} : \langle T \rangle.$
$\mathtt{C} \to \mathtt{B} : \{\mathsf{ok} : \mathtt{end}$
$\mathsf{quit} : \mathtt{end}\}$

$T = \oplus\langle \mathtt{S}, \{\mathsf{ok} : \ !\langle \mathtt{S}, \mathit{String} \rangle.?(\mathtt{S}, \mathit{Date}).\mathtt{end}, \ \mathsf{quit} : \ \mathtt{end}\rangle$

Example (delegation and processes)

$$\begin{aligned}
&\mathtt{A} \to \mathtt{S} : \langle \mathit{String} \rangle.\\
&\mathtt{S} \to \{\mathtt{A}, \mathtt{B}\} : \langle \mathit{Int} \rangle.\\
&\mathtt{A} \to \mathtt{B} : \langle \mathit{Int} \rangle.\\
&\mathtt{B} \to \mathtt{S} : \{\mathsf{ok} : \mathtt{B} \to \mathtt{S} : \langle \mathit{String} \rangle.\\
&\qquad\qquad \mathtt{S} \to \mathtt{B} : \langle \mathit{Date} \rangle.\mathtt{end},\\
&\qquad \mathsf{quit} : \mathtt{end}\}
\end{aligned}$$

$$\begin{aligned}
\mathtt{Seller} \quad = \quad & \overline{a}[\mathtt{S}](y).y?(\mathtt{A}, x_{title}).y!\langle \{\mathtt{A}, \mathtt{B}\}, \mathit{getPrice}(x_{title}) \rangle.\\
& y\&(\mathtt{B}, \{\mathsf{ok} : y?(\mathtt{B}, \mathit{address}).y!\langle \mathtt{B}, \mathit{today}() + 5 \rangle.\mathbf{0},\\
& \qquad\qquad \mathsf{quit} : \mathbf{0}\})
\end{aligned}$$

Example (delegation and processes)

$$\begin{aligned}
&\mathtt{A} \to \mathtt{S} : \langle \mathit{String} \rangle.\\
&\mathtt{S} \to \{\mathtt{A}, \mathtt{B}\} : \langle \mathit{Int} \rangle.\\
&\mathtt{A} \to \mathtt{B} : \langle \mathit{Int} \rangle.\\
&\mathtt{B} \to \mathtt{S} : \{\mathsf{ok} : \mathtt{B} \to \mathtt{S} : \langle \mathit{String} \rangle.\\
&\qquad\qquad \mathtt{S} \to \mathtt{B} : \langle \mathit{Date} \rangle.\mathtt{end},\\
&\qquad\quad \mathsf{quit} : \mathtt{end}\}
\end{aligned}$$

$$\begin{aligned}
\mathtt{Alice} \quad = \quad & a[\mathtt{A}](y).\\
& y!\langle \mathtt{S}, \text{“}\mathit{Flatland} : \mathit{A\ Romance\ of\ Many\ Dimensions}\text{”} \rangle.\\
& y?(\mathtt{S}, x_{quote}).\\
& y!\langle \mathtt{B}, x_{quote}\ \mathtt{div}\ 2 \rangle.\mathbf{0}
\end{aligned}$$

Example (delegation and processes)

$$\begin{aligned}
&\mathtt{A} \to \mathtt{S} : \langle \mathit{String} \rangle.\\
&\mathtt{S} \to \{\mathtt{A}, \mathtt{B}\} : \langle \mathit{Int} \rangle.\\
&\mathtt{A} \to \mathtt{B} : \langle \mathit{Int} \rangle.\\
&\mathtt{B} \to \{\mathtt{S}, \mathtt{A}\} : \{\mathsf{ok} : \mathtt{B} \to \mathtt{S} : \langle \mathit{String} \rangle.\\
&\qquad\qquad\qquad\qquad \mathtt{S} \to \mathtt{B} : \langle \mathit{Date} \rangle.\mathtt{end},\\
&\qquad\qquad\quad \mathsf{quit} : \mathtt{end}\}
\end{aligned}$$

$$\begin{aligned}
&\mathtt{B} \to \mathtt{C} : \langle \mathit{Int} \rangle.\\
&\mathtt{B} \to \mathtt{C} : \langle T \rangle.\\
&\mathtt{C} \to \mathtt{B} : \{\mathsf{ok} : \mathtt{end},\\
&\qquad\qquad\quad \mathsf{quit} : \mathtt{end}\}
\end{aligned}$$

$$\begin{aligned}
\mathtt{Bob} \quad = \quad & a[\mathtt{B}](y).y?(\mathtt{S}, x_{quote}).y?(\mathtt{A}, x_{contrib}).\\
& \mathtt{if}\ (x_{quote} - x_{contrib} < 100)\\
& \mathtt{then}\\
& \quad y \oplus \langle \{\mathtt{S}, \mathtt{A}\}, \mathsf{ok} \rangle.\\
& \quad y!\langle \mathtt{S}, \text{“}\mathit{Via\ dei\ matti\ 0}\text{”} \rangle.\\
& \quad y?(\mathtt{S}, x_{date}).\mathbf{0}\\
& \mathtt{else}\\
& \quad b[\mathtt{B}](z).z!\langle \mathtt{C}, x_{quote} - x_{contribution} - 99 \rangle.\\
& \quad z!\langle\langle \mathtt{C}, y \rangle\rangle.\\
& \quad z\&(\mathtt{C}, \{\mathsf{ok} : \mathbf{0},\ \mathsf{quit} : \mathbf{0}\})
\end{aligned}$$

Example (delegation and processes)

$$\begin{aligned}
&\mathtt{B} \to \mathtt{C} : \langle Int \rangle.\\
&\mathtt{B} \to \mathtt{C} : \langle T \rangle.\\
&\mathtt{C} \to \mathtt{B} : \{\mathsf{ok} : \mathtt{end},\\
&\qquad\qquad \mathsf{quit} : \mathtt{end}\}
\end{aligned}$$

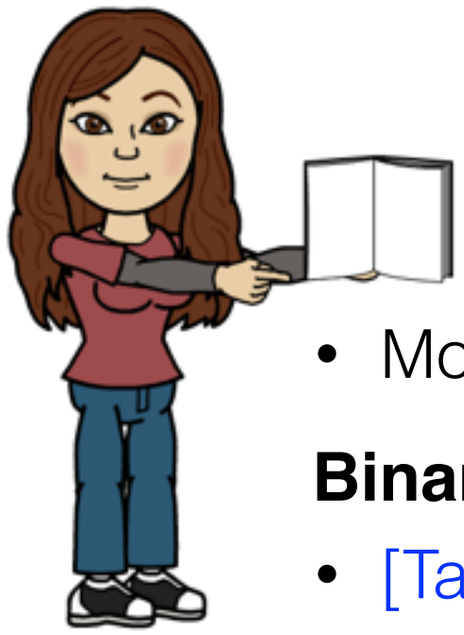
$$T = \oplus\langle\{\mathtt{S}, \mathtt{A}\}, \{\mathsf{ok} : !\langle \mathtt{S}, \mathit{string}\rangle.?(\mathtt{S}, \mathit{date}).\mathtt{end},\ \ \mathsf{quit} : \mathtt{end}\rangle$$

$$\begin{aligned}
\mathtt{Carol} \quad = \quad & b[\mathtt{C}](z).z?(\mathtt{B}, x).z?((\mathtt{B}, t)).\\
& \mathtt{if}\ (x < 100)\\
& \mathtt{then}\\
& \quad z \oplus \langle \mathtt{B}, \mathsf{ok}\rangle.t \oplus \langle\{\mathtt{S}, \mathtt{A}\}, \mathsf{ok}\rangle.\\
& \quad t!\langle \mathtt{S}, \text{“}28^{th}\ October\ \&\ Makarios\ III\ Ave, Cyprus''\rangle.\\
& \quad t?(\mathtt{S}, x_{date}).\mathbf{0}\\
& \mathtt{else}\\
& \quad z \oplus \langle \mathtt{B}, \mathsf{quit}\rangle.t \oplus \langle\{\mathtt{S}, \mathtt{A}\}, \mathsf{quit}\rangle.\mathbf{0}
\end{aligned}$$

Typing delegation

[Deleg]
$$\frac{\Gamma \vdash P \triangleright \Delta, c : T}{\Gamma \vdash c!\langle\langle \mathtt{p}, c' \rangle\rangle.P \triangleright \Delta, c :!\langle \mathtt{p}, \mathsf{T} \rangle, c' : \mathsf{T}}$$

[Srcv]
$$\frac{\Gamma \vdash P \triangleright \Delta, c : T, y : \mathsf{T}}{\Gamma \vdash c?((\mathtt{p}, y)).P \triangleright \Delta, c :!(\mathtt{p}, \mathsf{T})}$$



References

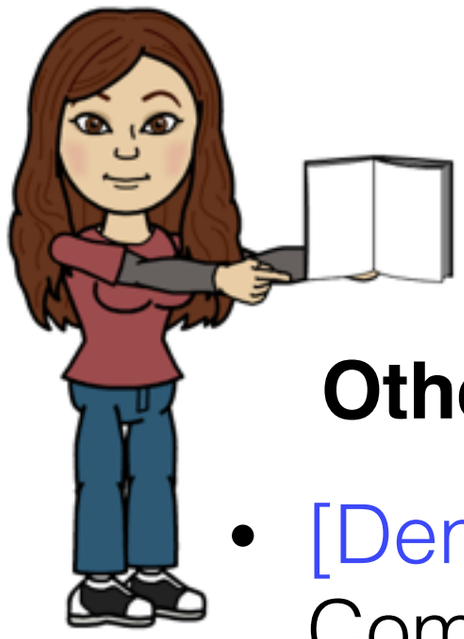
- Mobility Reading Group's home page **http://mrg.doc.ic.ac.uk**

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@ICALP'13] Multiparty Compatibility in Communicating Automata: Characterisation and Synthesis of Global Session Types
- [Deniélou,Yoshida@POPL'11] Dynamic Multirole Session Types.
- [Beljeri,Yoshida@PLACES'08] Synchronous Multiparty Session types
- [Kouzapas,Yoshida@CONCUR'13] Globally Governed Session Semantics