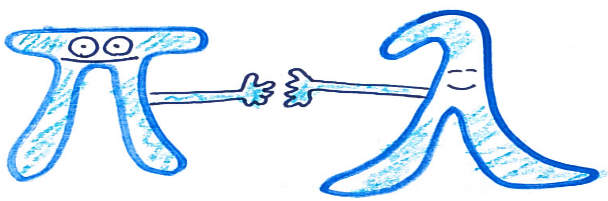
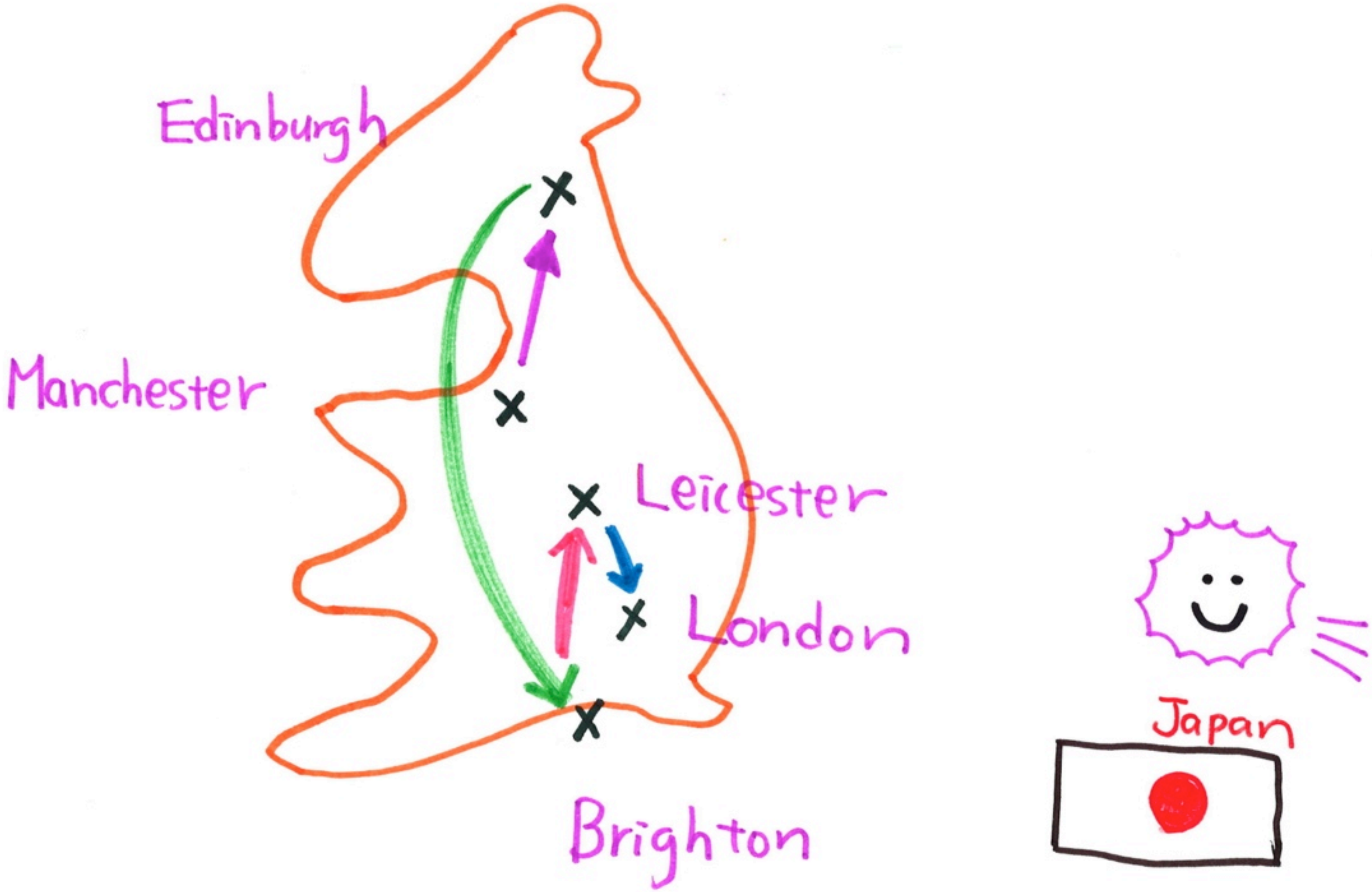
HOπ and TYPES



NOBUKO YOSHIDA

My Mobile Research Life in UK



1998 May – 1999 Sep

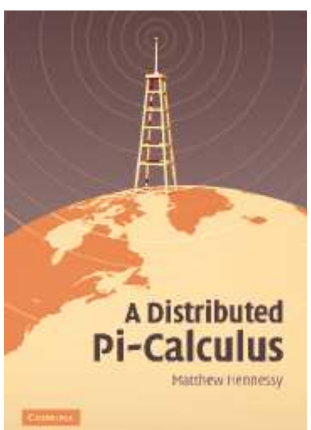
Before starting a post-doc position ...

- John Power @ Edinburgh

Matthew is **ROYAL**

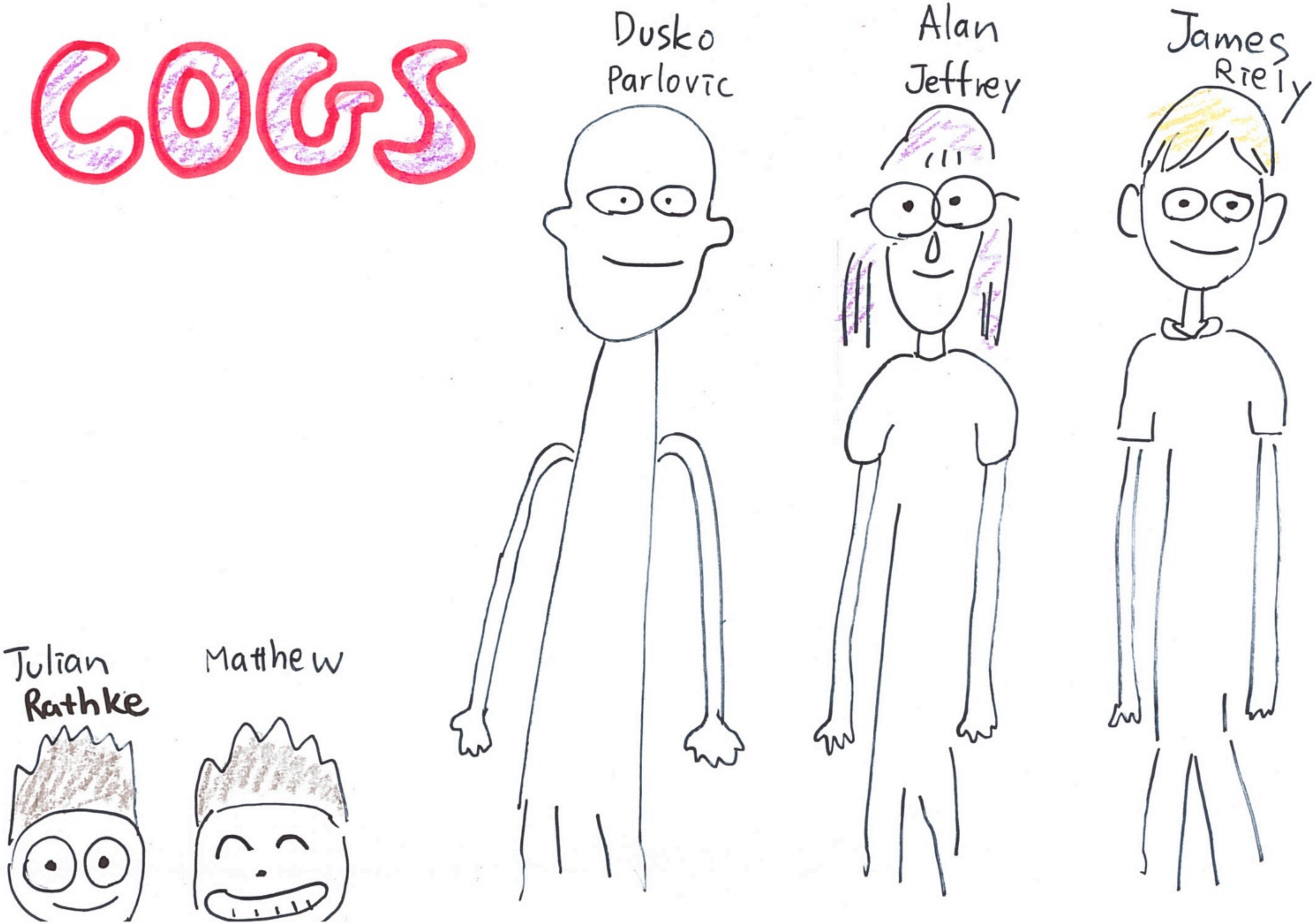
- TOPICS

Dπ



* Subtyping

* HOπ


COGS
Dusko
Parlovic
Alan
Jeffrey
James
Riely
Julian
Rathke
Matthew

- ❖ **Assigning Types to Processes** [LICS 2000, I&C]

Matthew Hennessy, NY

- ❖ **SafeDpi**: A Language for Controlling Mobile Code [FoSSaCS 2004, Acta Informatica]

Matthew Hennessy, Julian Rathke, NY

Higher-Order π-Calculus

[Sangiorgi 93]

CML, Facile, LLinda, ...

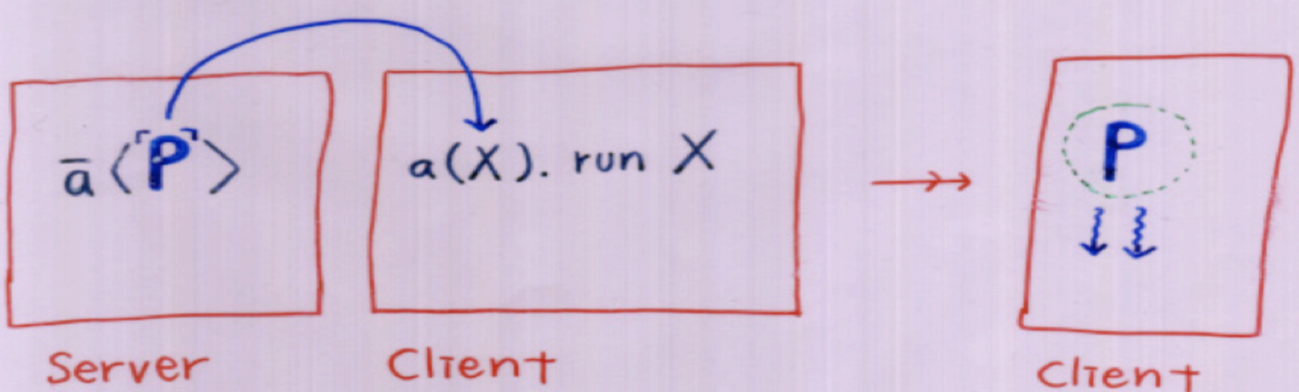
λ_v $\quad (\lambda x.Q)V \longrightarrow Q[V/x]$

$\lambda\pi_v$ $\quad \bar{a}\langle \underset{\text{code}}{\ulcorner P \urcorner} \rangle \mid a(X).\,\mathrm{run}\,X$

$\quad \longrightarrow \quad \mathrm{run}\ulcorner P \urcorner \longrightarrow P$

where $\quad \underset{\text{thunk}}{\ulcorner P \urcorner} = \lambda().\,P$

$\quad \mathrm{run} = \lambda X.\,X()$

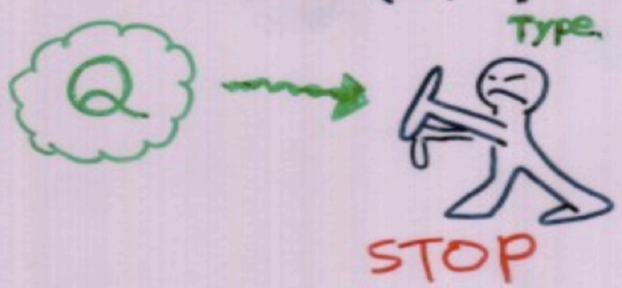


Aims of Types / Typechecking

- Using Types to control the effects of Mobile Code / Processes

- Host refuses to execute incoming code unless it conforms to predetermined access policy

$$a(x : \mathbf{T}).\,P$$



Existing λ/π Typing System

[1993 – 2000]

Before [YH00]

λ→ + πIO

$$\tau \;::=\; \mathrm{unit} \;\mid\; \mathrm{nat} \;\mid\; \tau \to \rho \;\mid\; \sigma \;\mid\; \mathrm{proc}$$

Value Term

constant process type

$$\sigma \;::=\; (\tilde{\tau})^{I} \;\mid\; (\tilde{\tau})^{O} \;\mid\; (\tilde{\tau})^{IO}$$

channel — Input — Output — Input-Output

Typing Processes .... Too Simple

$$\boxed{\Gamma \vdash P : \mathrm{proc}}$$

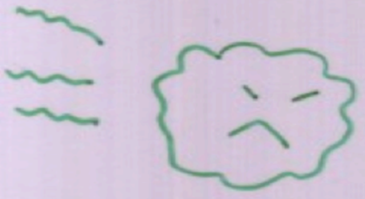
e.g. $$\frac{\Gamma \vdash P : \mathrm{proc} \qquad \Gamma \vdash Q : \mathrm{proc}}{\Gamma \vdash P \mid Q : \mathrm{proc}}$$

cf. $\Gamma \vdash M : \tau \to \rho$

Problem

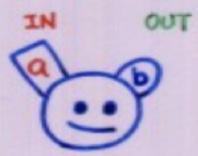
$$c(X : \ulcorner \mathrm{proc} \urcorner).\ \mathrm{run}\ X$$

where $\ulcorner \mathrm{proc} \urcorner = \mathrm{unit} \rightarrow \mathrm{proc}$

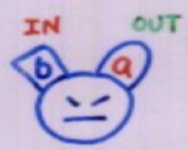


Any Process is welcome

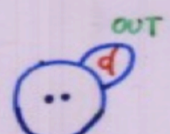
P $\underline{a}(X).\ \underline{\bar{b}}\langle X \rangle$



Q $\underline{b}(X).\ \underline{\bar{a}}\langle X \rangle$



R $\underline{d}\langle \mathrm{string} \rangle$



We are very <u>different</u>

the Idea is simple but ...

quite a bit of work

[1998 – 2000 – 2003]

Channels appear both in Types and Processes

$$F = \lambda x.\lambda(X : \ulcorner x{:}(\tau)^{\circ}, b{:}(\tau)^{\circ}\urcorner).(\mathrm{run}\ X \mid \overline{x}\langle 1\rangle \mid \overline{b}\langle 2\rangle)$$

$$F\,a \rightarrow \lambda(X : \ulcorner a{:}(\tau)^{\circ}, b{:}(\tau)^{\circ}\urcorner).(\mathrm{run}\ X \mid \overline{a}\langle 1\rangle \mid \overline{b}\langle 2\rangle)$$

$$F\,b \rightarrow \lambda(X : \ulcorner b{:}(\tau)^{\circ}\urcorner).(\mathrm{run}\ X \mid \overline{b}\langle 1\rangle \mid \overline{b}\langle 2\rangle)$$

⇒ Kinding / Dependency Types

[Yoshida · Hennessy 2000] • Functional Channel Dependency

Another bit of work

New • Channel Dependency (Non-determinism)

New • Existential Types (Scope-Opening)

Higher Order π-calculus

λπv

Syntax

$$
\begin{aligned}
P, Q ::= \;& V && \text{value} \\
\mid\;& 0 && \text{nil} \\
\mid\;& P \mid Q && \text{parallel} \\
\mid\;& \bar{u}\langle V_1, \dots, V_n \rangle && \text{OUTPUT} \\
\mid\;& u(X_1{:}\tau_1, \dots, X_n{:}\tau_n)P && \text{INPUT} \\
\mid\;& !P && \text{Replication} \\
\mid\;& (\nu a{:}\sigma)P && \text{Restriction} \\
\mid\;& PQ && \text{Application} \\
V, W ::= \;& \lambda(x{:}\tau)P && \lambda\text{-abst} \\
\mid\;& 1, 2, \dots, (), \dots && \text{constant} \\
\mid\;& x, y, z, \dots && \text{variables} \\
\mid\;& a, b, c, \dots && \text{channels/names}
\end{aligned}
$$

Types

$$
\begin{array}{lllll}
\text{Term} & \tau & ::= & \mathrm{unit},\ \mathrm{nat} & \\
 & & \mid & \tau \to \tau' & \\
 & & \mid & \Pi(x:\sigma)\tau & \text{functional dependency} \\
 & & \mid & [\Delta] & \text{process} \\
 & & \mid & \sigma & \\
 & & & & \\
\text{Channel} & \sigma & ::= & (\Pi[\tilde{x}:\tilde{\sigma}]\tilde{\tau})^{\rho} & \text{channel dependency} \\
 & & \mid & (\exists[\tilde{x}:\tilde{\sigma}]\tilde{\tau})^{\rho} & \text{existential} \\
 & & \mid & \langle \sigma_{I}, \sigma_{O} \rangle & \\
 & & & & \\
 & \rho & ::= & I \mid O &
\end{array}
$$

Typing System $\Gamma \vdash P \triangleright I$

(Zero) $$\frac{\Gamma \vdash Env}{\Gamma \vdash 0 \triangleright [\,]}$$

(Par) $$\frac{\Gamma \vdash P \triangleright [\Delta] \quad \Gamma \vdash Q \triangleright [\Delta']}{\Gamma \vdash P \mid Q \triangleright [\Delta \cdot \cup \Delta']}$$

(Res) $$\frac{\Gamma, a{:}6 \vdash P \triangleright [\Delta, a{:}6]}{\Gamma \vdash (\nu a{:}6)\, P \triangleright [\Delta]}$$

(Rep) $$\frac{\Gamma \vdash P \triangleright [\Delta]}{\Gamma \vdash !P \triangleright [\Delta]}$$

Examples

FW(ab) $\Gamma \vdash \underline{a}(x:\tau).\,\bar{b}\langle x\rangle : [\underline{a}:(\tau)^{I},\ b:(\tau)^{O}]$

FW(ba) $\Gamma \vdash b(x:\tau).\,\bar{\underline{a}}\langle x\rangle : [b:(\tau)^{I},\ \underline{a}:(\tau)^{O}]$

par $\Gamma \vdash FW(ab) \mid FW(ba) : [a:(\tau)^{IO},\ b:(\tau)^{IO}]$

because

$$[a:(\tau)^{I},\ b:(\tau)^{O}] \sqcup [a:(\tau)^{O},\ b:(\tau)^{I}]$$

$$= [a:(\tau)^{IO},\ b:(\tau)^{IO}]$$

restriction $a:(\tau)^{IO},\ b:(\tau)^{IO} \vdash FW(ab) : [a:(\tau)^{I},\ \underline{b}:(\tau)^{O}]$

$$\Downarrow$$

$$a:(\tau)^{IO} \vdash (\nu \underline{b})\, FW(ab) : [\underline{a}:(\tau)^{I}]$$

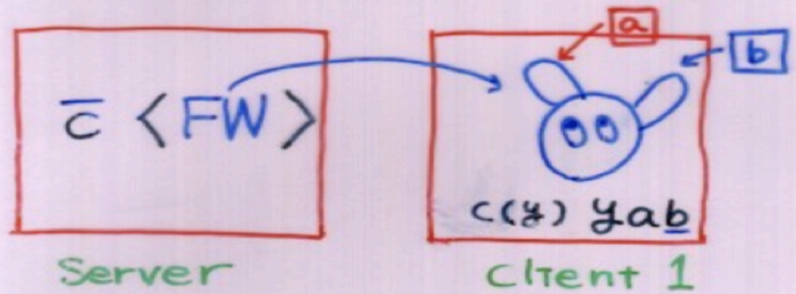
$$\Downarrow$$

$$\vdash (\nu \underline{a} b)\, FW(ab) : [\,]$$

Script Server

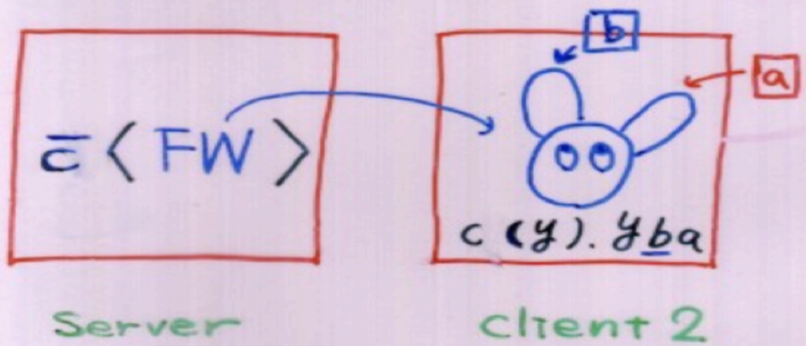
$$FW = \lambda(x:\tau).\lambda(y:\tau).\ x(z).\overline{y}\langle z\rangle$$

forwarder



C1 evolves into

$FW(ab)$
$: [a:(\tau)^I,\ b:(\tau)^O]$



C2 evolves into

$FW(ba)$
$: [a:(\tau)^O,\ b:(\tau)^I]$

Previous $\quad (\tau)^I \rightarrow (\tau)^O \rightarrow \text{proc}$

Func Dep
[YHOO]

$$\Pi(\underline{x}:(\tau)^I)\ \Pi(\underline{y}:(\tau)^O)\ [x:(\tau)^I,\ y:(\tau)^O]$$

Existential Types for Scope Opening

Client (A) wishes to execute code P and to get ack $\bar{c}\langle V \rangle$ at the time P is executed at the remote location (B)

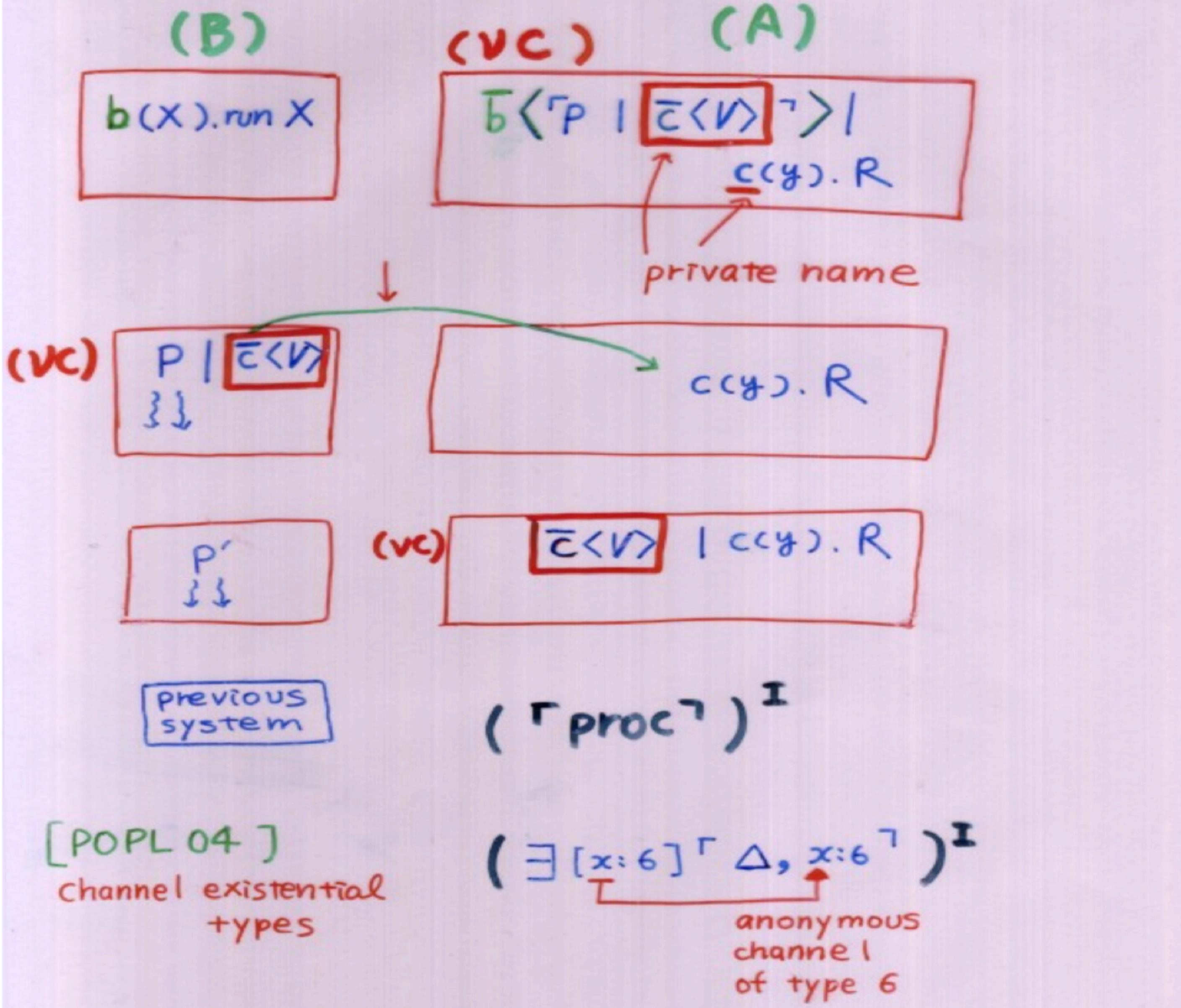
Location (A)

$$(\nu c)\ \Big(\ \underbrace{\bar{b}\langle \ulcorner P}_{\text{code}}\ |\ \underbrace{\bar{c}\langle V\rangle \urcorner}_{\text{ack}}\rangle\ \Big|\ \underbrace{c(x).R}_{\text{wait at } c}\ \Big)$$

(1) c is private

(2) V must not be touched (i.e. compromised)

Existential Types for Scope Opening



Typing System for $\exists$

$$(\mathrm{In}^{\exists})\quad \frac{\Gamma \vdash \underline{a} : (\exists[x:\sigma]\tau)^{I} \qquad \Gamma, \overset{\text{pack}}{\{x:\sigma, X:\tau\}} \vdash P \triangleright [\Delta, x:\sigma]}{\Gamma \vdash a(X : \exists[x:\sigma]\tau).P \triangleright [\Delta, \underline{a} : (\exists[x:\sigma]\tau)^{I}]}$$

$$(\mathrm{Out}^{\exists})\quad \frac{\Gamma \vdash a : (\exists[x:\bar{\sigma}]\tau)^{O} \qquad \Gamma \vdash \overset{\text{pack}}{\{c, V\}} : \exists[x:\sigma]\tau}{\Gamma \vdash \bar{a}\langle V\rangle \triangleright [a : (\exists[x:\bar{\sigma}]\tau)^{O}, \underline{c:\sigma}]}$$

record a name
to be restricted

Proposition (Minimum Interface)

$$\Gamma \vdash P \triangleright [\Delta_i] \Rightarrow \exists!\, \Delta_{min} \subseteq \Delta_i$$
$$\text{s.t. } \Gamma \vdash P \triangleright [\Delta_{min}]$$

Main Theorems

Subject Reduction

$$\Gamma \vdash P : \tau,\ P \twoheadrightarrow P' \Rightarrow \Gamma \vdash P' : \tau$$

Type Safety

$$\Gamma \vdash P : [\Delta] \Rightarrow \quad P \not\xrightarrow{\Gamma,\,[\Delta]}_{err}$$

where $P \not\xrightarrow{\Gamma,\,[\Delta]}_{err}$ means

P can use **at most** resources in Δ

Consequence:

$$a(X : \ulcorner \Delta \urcorner).P \mid \overline{a}\langle \ulcorner R \urcorner \rangle \xrightarrow{\Gamma,\,\pi}_{err}$$

if $\Gamma \nvdash R : [\Delta]$

Encapsulation of Higher-Order Code by Hidden Name

Theorem

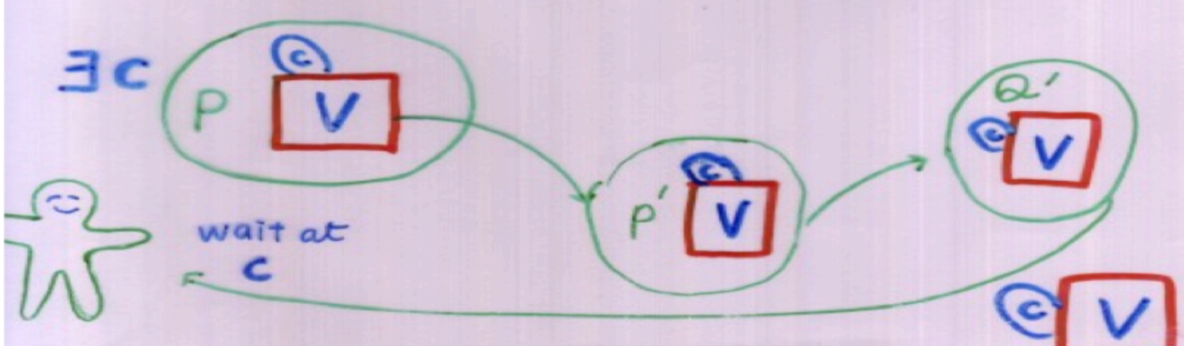
$\Gamma \vdash P : [\Delta]$ and $fv(P) = \emptyset$

and $\Gamma \vdash a : (\exists (x : \sigma^{\uparrow}) \ulcorner x : \sigma^{\uparrow} \urcorner)^{!}$

linear name

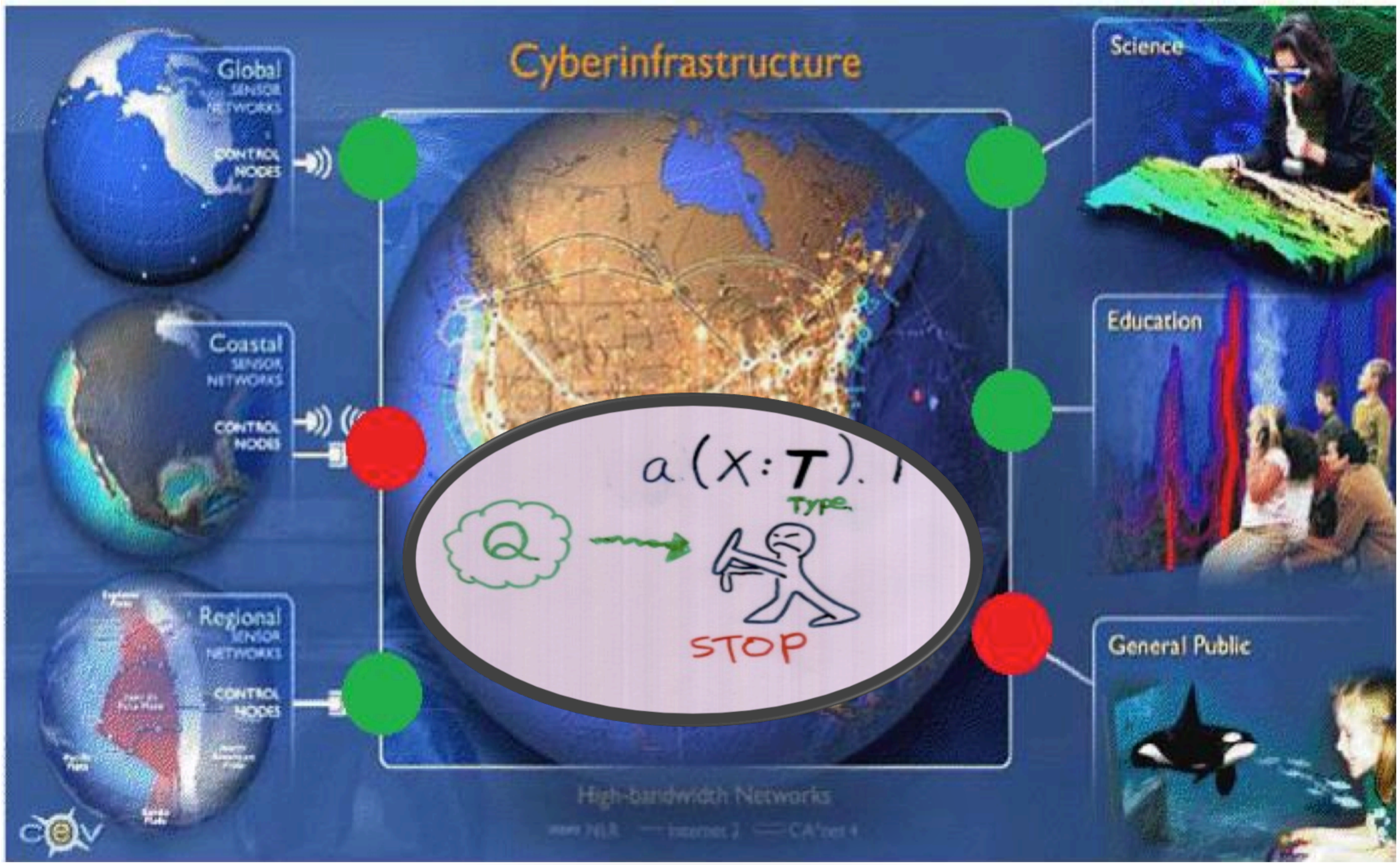
Then $P \xrightarrow{a\langle \ulcorner \overline{c}\langle V \rangle \urcorner \rangle} P' \Rightarrow P' \twoheadrightarrow \xrightarrow{\overline{c}\langle V \rangle}$

Mobile Code bound by $\exists$-name is eventually returned to the sender without being touched by the receiver



Ocean Observatory Initiative

a use of multiparty session types as a runtime monitor



ESOP 2009: Asynchronous Subtyping for Session Types [Mostrous, Honda, NY]

P_1

$t?(y);\ s!\langle 5\rangle;\ s!\langle \text{Apple}\rangle$

3

P_3

$s?(z_1);\ s?(z_2);$

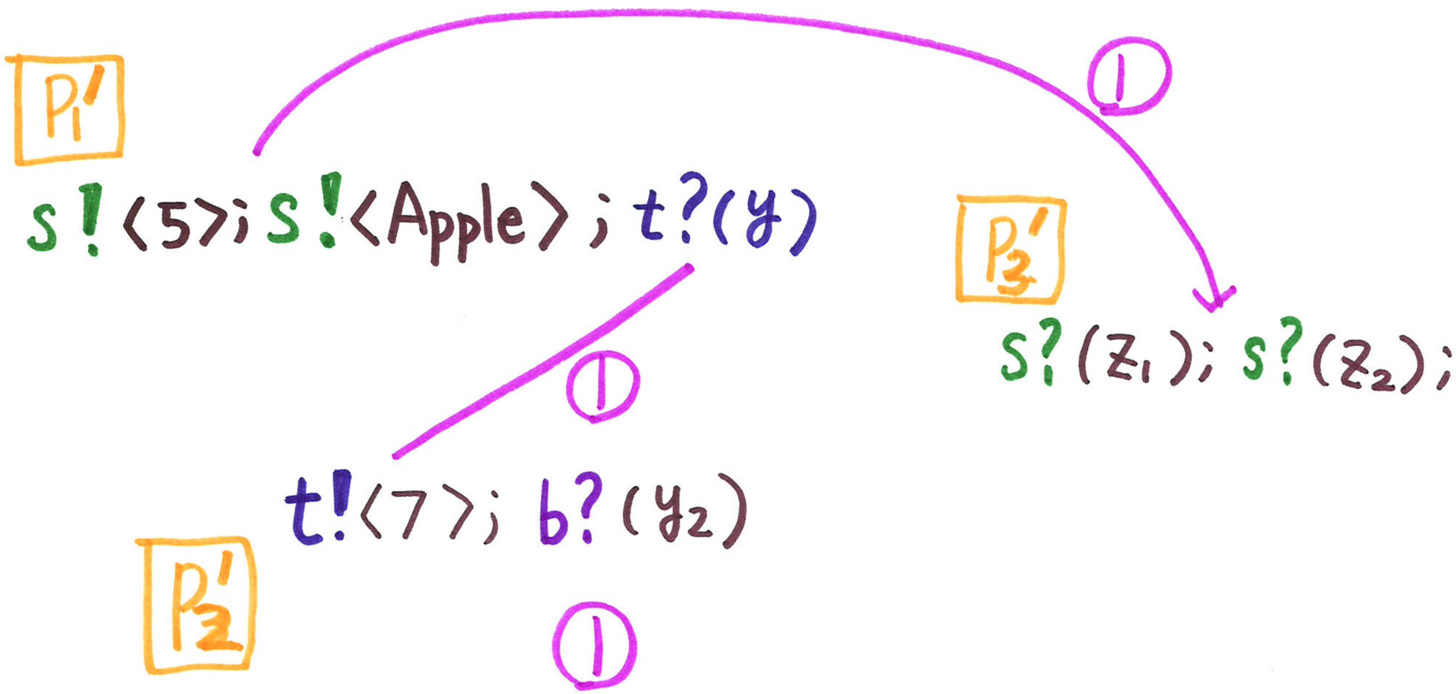
2

P_2

$b?(y_2);\ t!\langle 7\rangle$

1

block

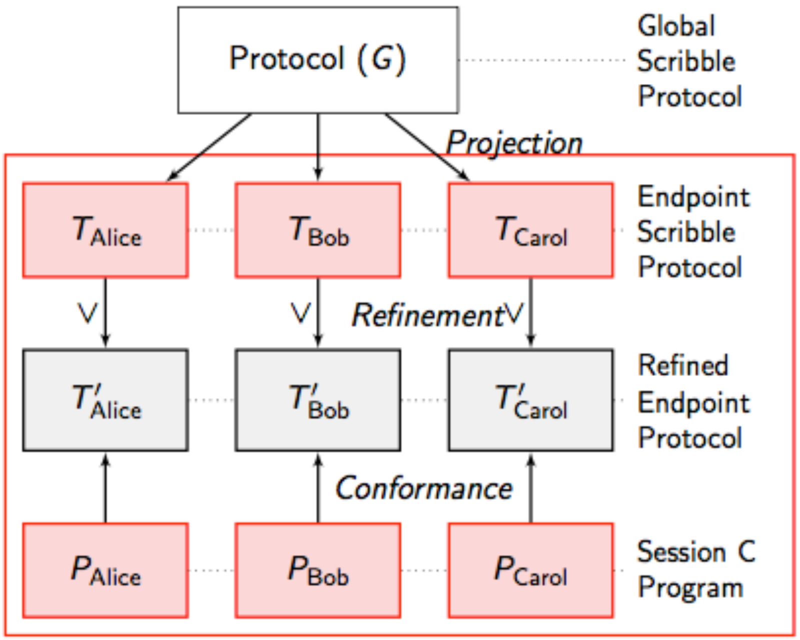


Communication Subtyping $!\langle T\rangle; ?\langle T'\rangle \leqslant ?\langle T'\rangle; !\langle T\rangle$

PPDP'14: Preciseness of Session Subtyping
[Tzu-Chun Chen, Mariangiola Dezani and NY]

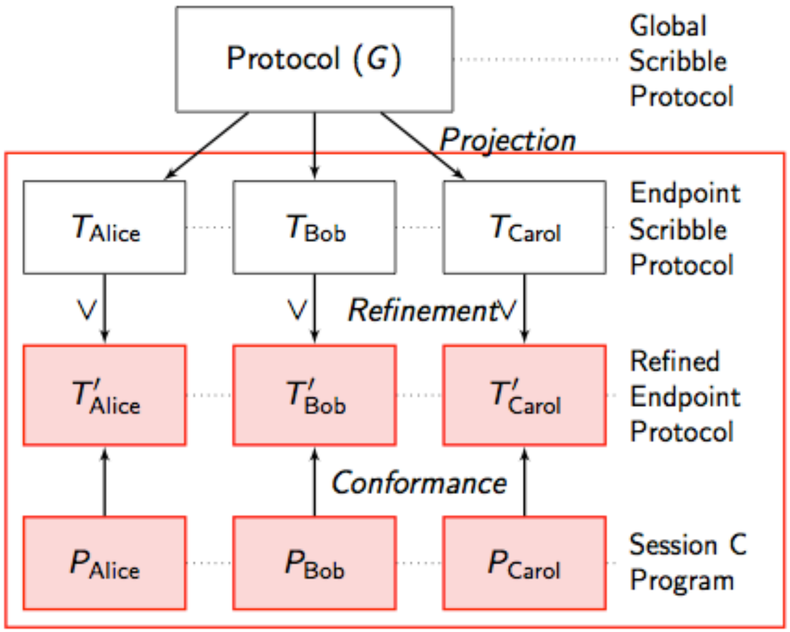
Session Type checking

- Static analyser
- Does source code conform to specification?
- Extract session type from code
  - Based on usage of API
  - Based on program flow control
- Compare w/ endpoint protocol



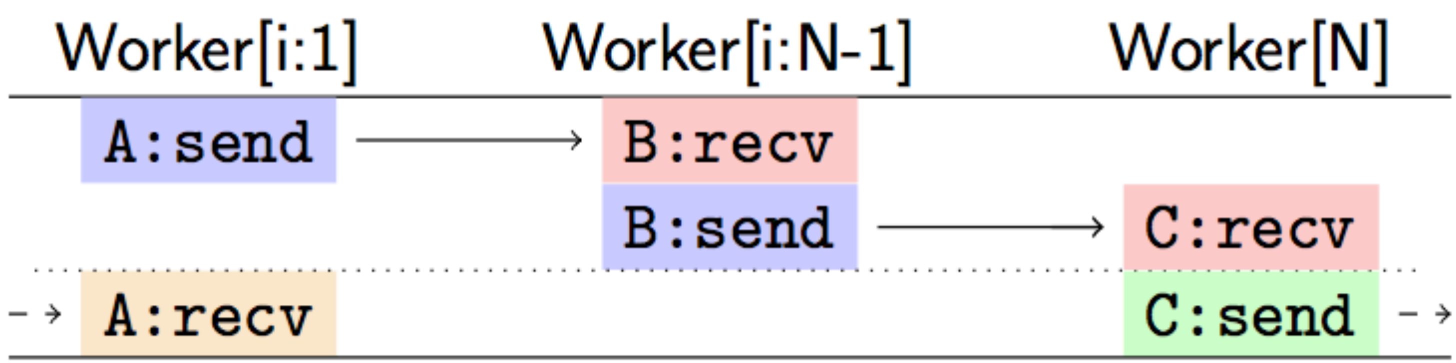
Session Type checking: Asynchronous optimisation

- Protocols designed safe
- Naive impl. inefficient
- Asynchronous impl.
  - Non-blocking send
  - Blocking receive
- Overlap send/recv operations
- Safety by async. subtyping [Mostrous et al., ESOP'09]



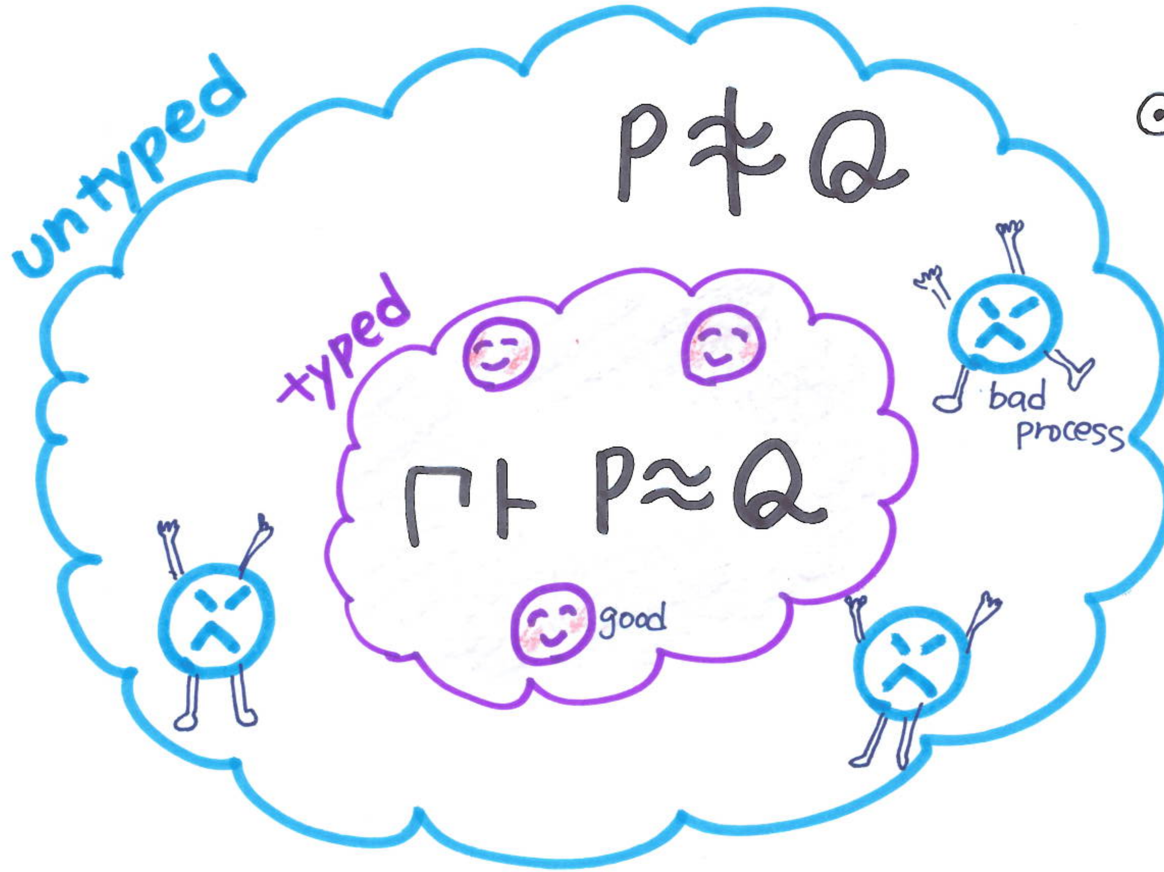
Example: Ring topology in Pabble

```
global protocol Ring(role Worker[1..N]) {
  rec LOOP {
    Data(int) from Worker[i:1..N-1] to Worker[i+1] ;
    Data(int) from Worker[N] to Worker[1] ;
    continue LOOP;
  }
}
```



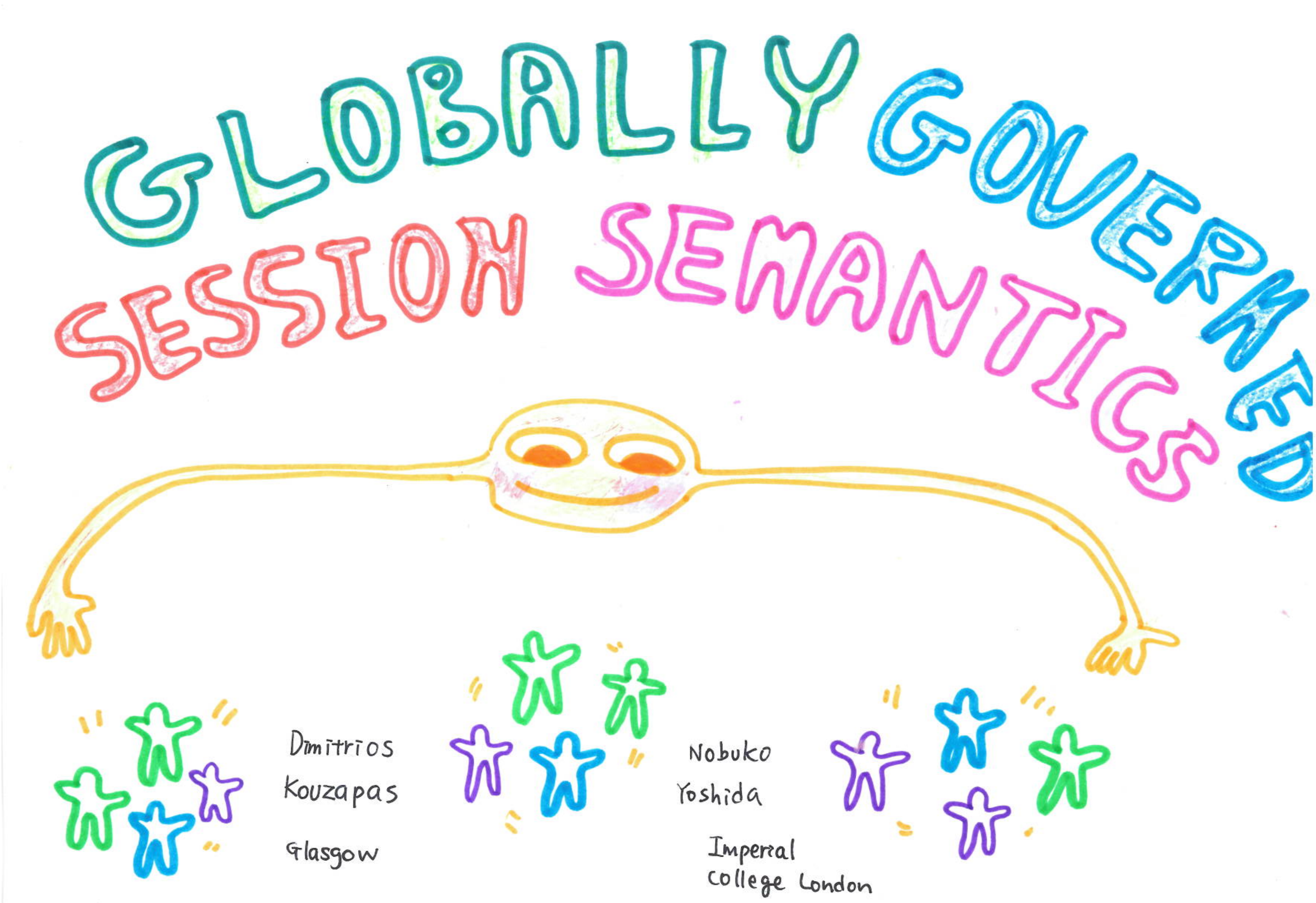
Typed Semantics in π 1991 →

IO-subtyping, Linear types, Secure Information Flow, ...



- Correctness of Encoding
- Limit environments Γ
  ⇒ Equate more processes
- Compositional

Many Hennessy's papers study typed/environment bisimulation


GLOBALLY GOVERNED
SESSION SEMANTICS
Dmitrios
Kouzapas
Glasgow
Nobuko
Yoshida
Imperial
college London

Yuxin Deng, Matthew Hennessy, ICALP 2011 =>

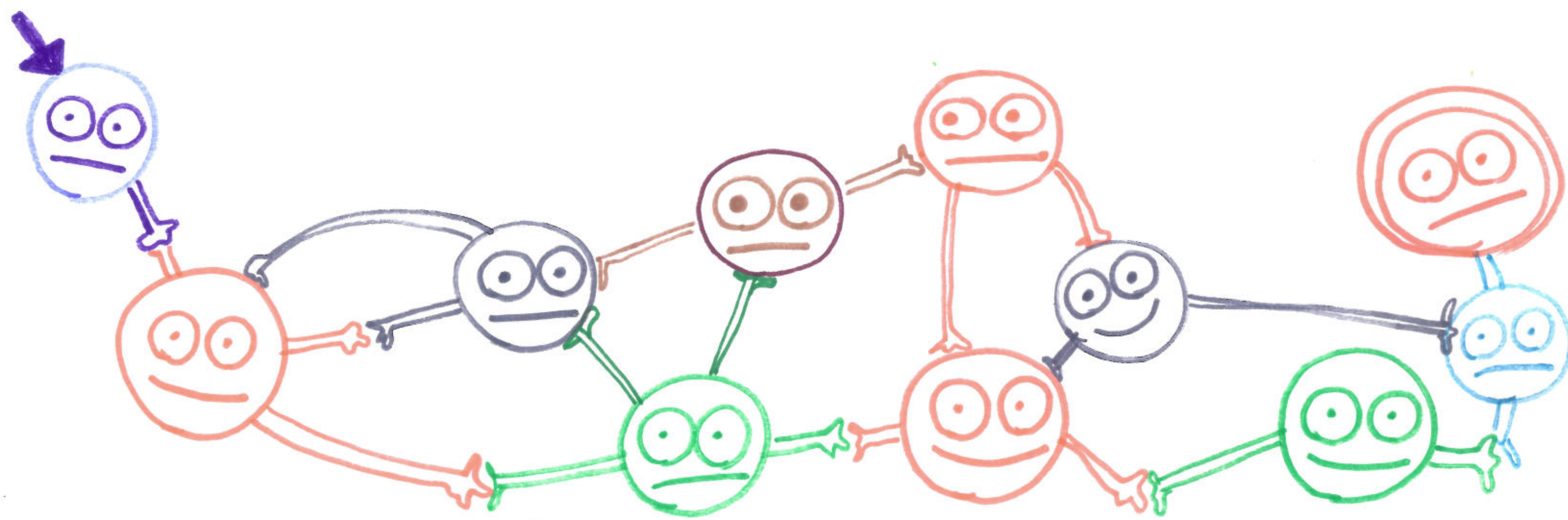
Multiparty Compatibility in Communicating Automata

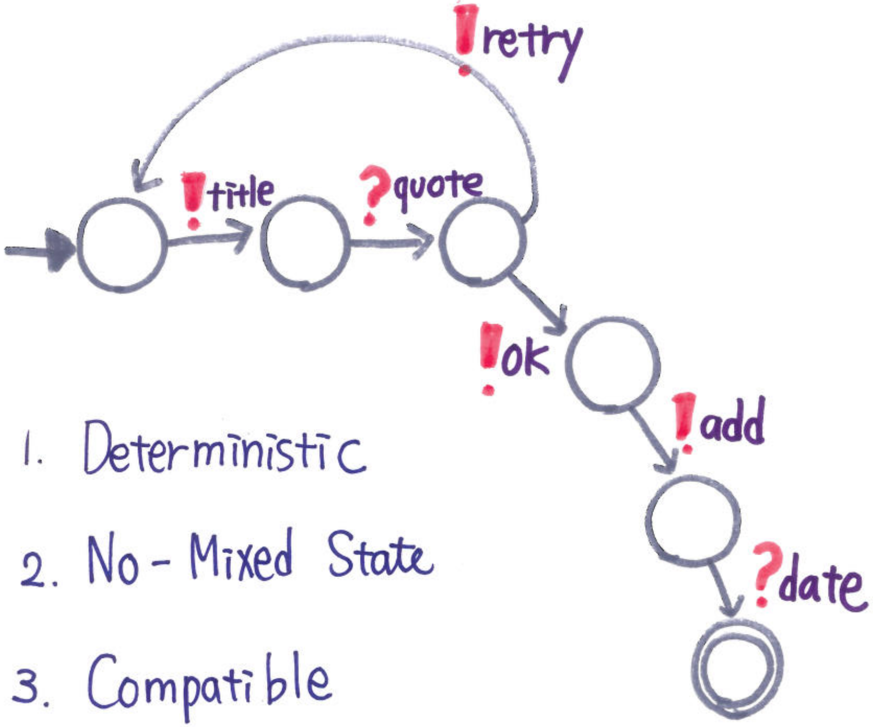
Synthesis and Characterisation of Multiparty Session Types

ICALP'13

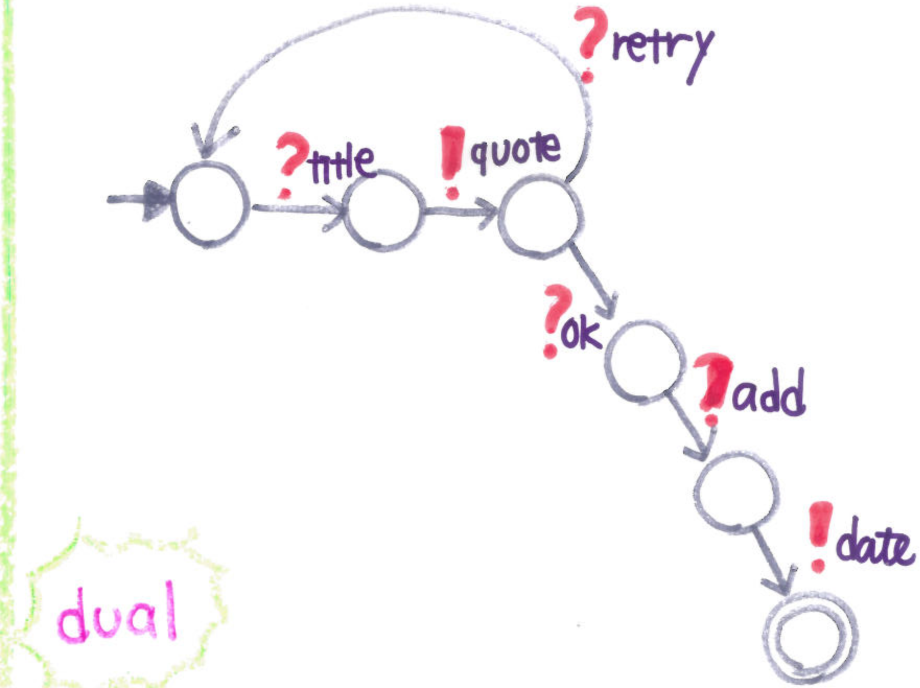
Nobuko Yoshida

Pierre-Malo Deniélou





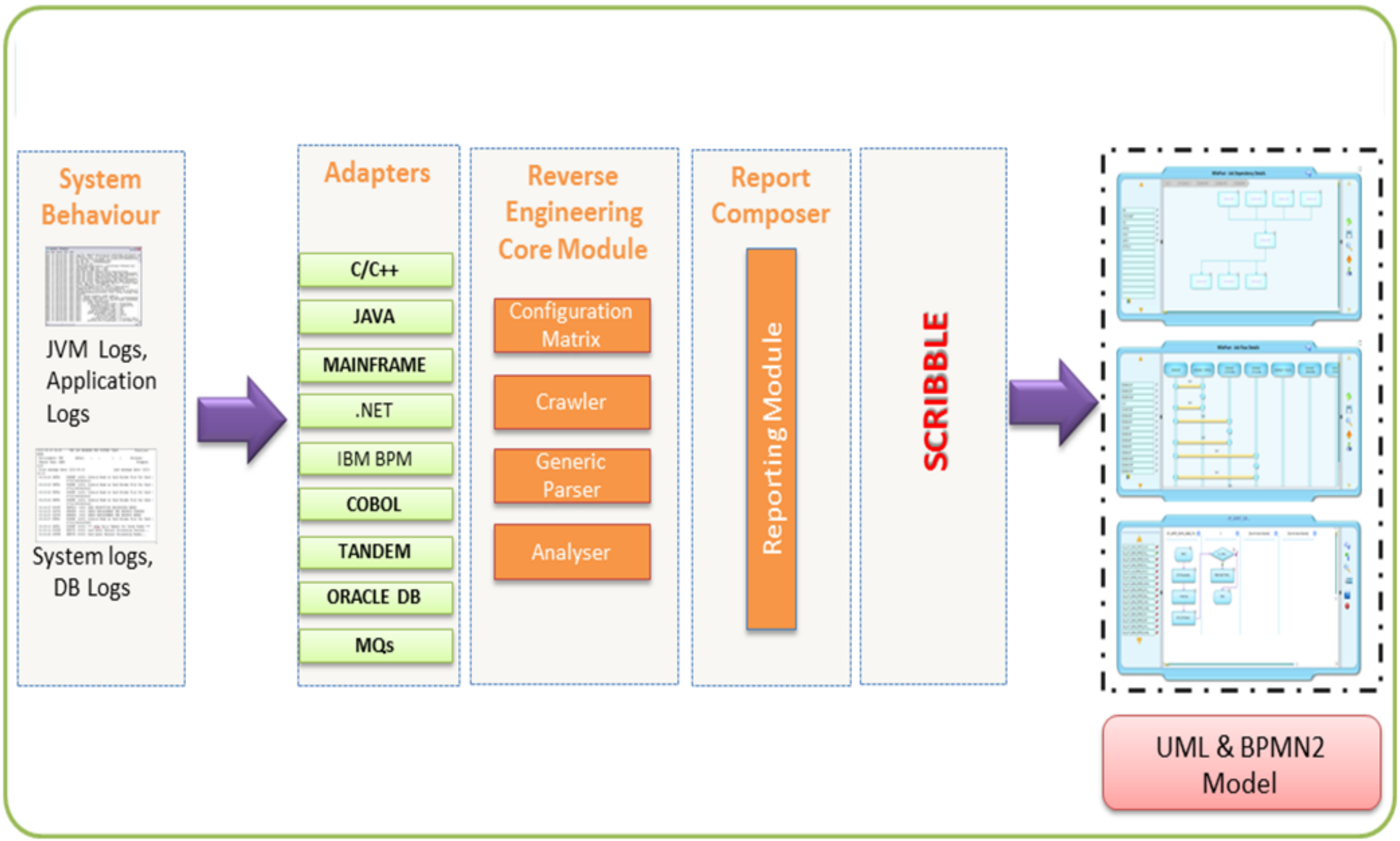
1. Deterministic
2. No-Mixed State
3. Compatible



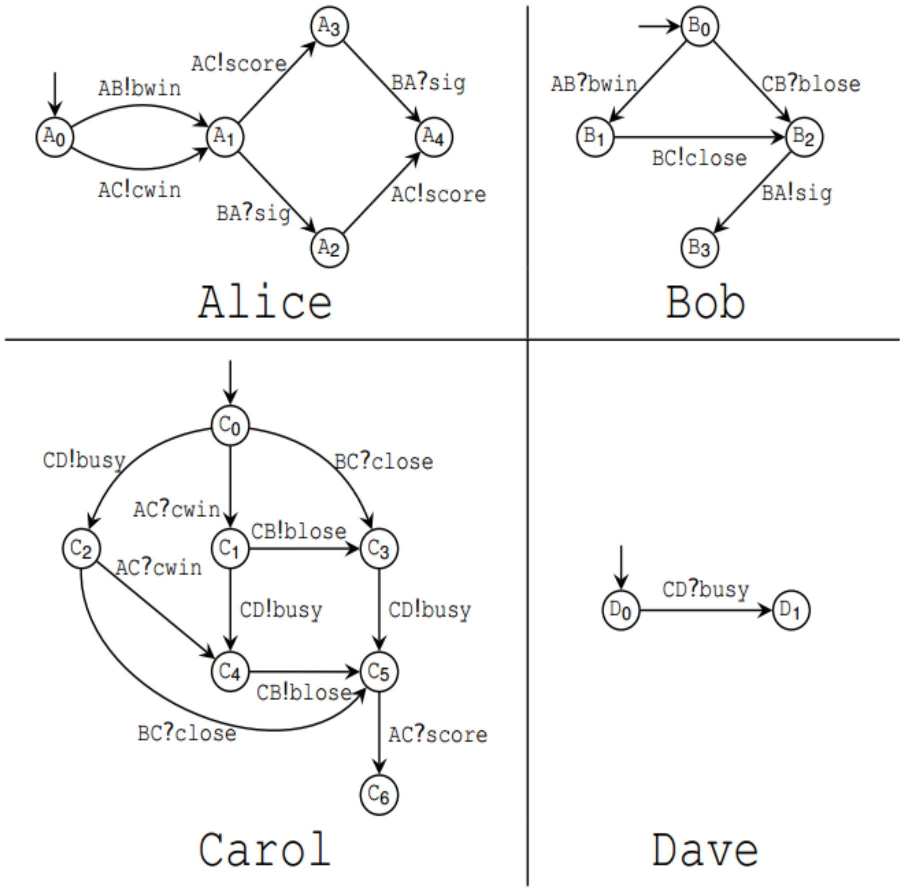
[Gouda et al 1986] Two compatible machines without mixed states which are deterministic satisfy deadlock-freedom.

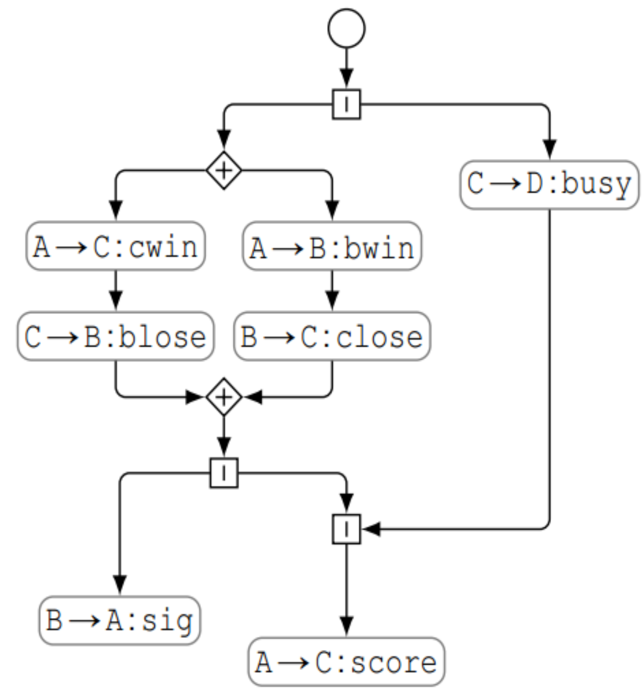
http://www.zdlc.co/faq/

Zero Deviation Life Cycle Platform



POPL 2015: From Communicating Machines to Graphical Choreographies [Lange, Tuosto, NY]

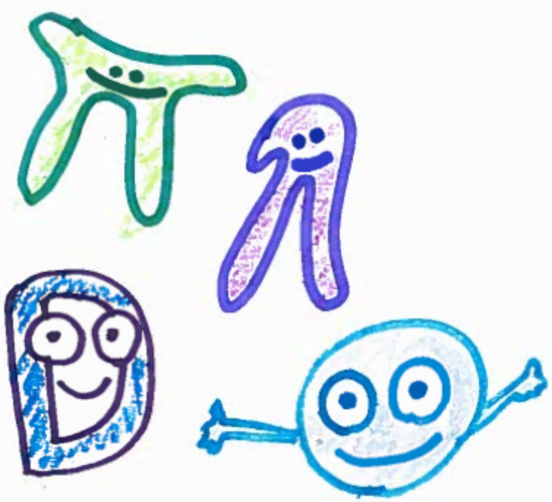




HAPPY

BIRTH

DAY



Matthew