A Reputation System for Multirole Sessions

Mariangiola Dezani

Dipartimento di Informatica
Università di Torino

joint work with Viviana Bono and Sara Capecchi and Ilaria Castellani

TGC 2011 - Aachen

Scenario

communication protocols with many roles and a varying number of principals playing different roles;

Scenario

communication protocols with many roles and a varying number of principals playing different roles;

Extension

- policies for services and principals;
- objective reputations of principals;

Scenario

communication protocols with many roles and a varying number of principals playing different roles;

Extension

- policies for services and principals;
- objective reputations of principals;

Contribution

a core calculus equipped with a type system assuring communication safety and progress.

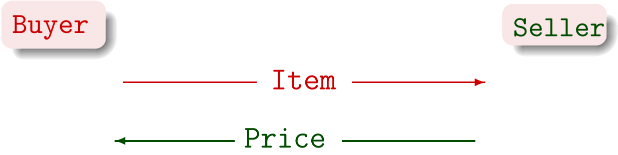
Diadic Sessions

Buyer

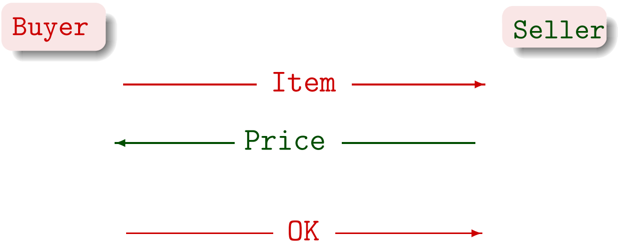
Seller

Diadic Sessions

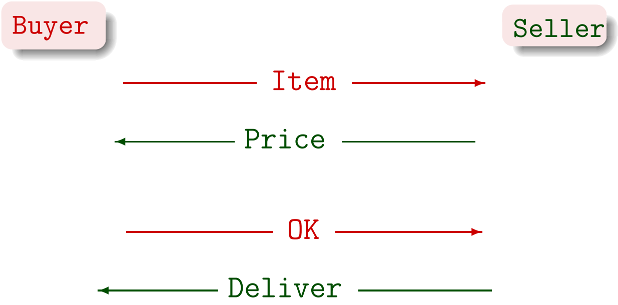
Diadic Sessions



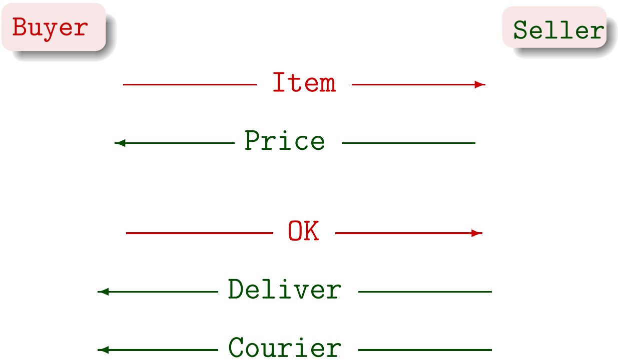
Diadic Sessions



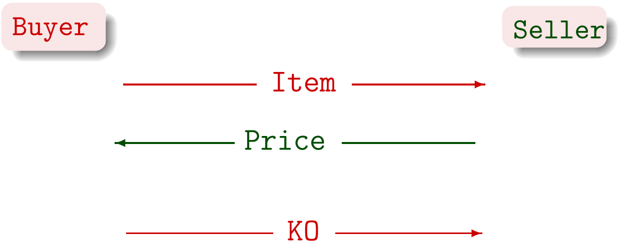
Diadic Sessions



Diadic Sessions



Diadic Sessions



Diadic Sessions

Kohei Honda. Types for Dyadic Interaction. In *Proc. CONCUR'93*, volume 715 of *LNCS*, pages 509–523. Springer, 1993.

Kaku Takeuchi, Kohei Honda, and Makoto Kubo. An Interaction-based Language and its Typing System. In *PARLE'94*, volume 817 of *LNCS*, pages 398–413. Springer, 1994.

Kohei Honda, Vasco Vasconcelos, and Makoto Kubo. Language Primitives and Type Disciplines for Structured Communication-based Programming. In *ESOP'98*, volume 1381 of *LNCS*, pages 22–138. Springer, 1998.

Global Type

$$
\begin{array}{lll}
\texttt{Buyer} \rightarrow \texttt{Seller} & \langle \texttt{Item} \rangle ; \\
\texttt{Seller} \rightarrow \texttt{Buyer} & \langle \texttt{Price} \rangle ; \\
\texttt{Buyer} \rightarrow \texttt{Seller} & \{ \texttt{OK}.\ \texttt{Seller} \rightarrow \texttt{Buyer} \langle \texttt{Deliver} \rangle ; \\
 & \qquad \texttt{Seller} \rightarrow \texttt{Buyer} \langle \texttt{Courier} \rangle ; \mathsf{end}, \\
 & \ \texttt{KO}.\ \mathsf{end} \\
 & \}
\end{array}
$$

Global Type

$$
\begin{array}{lll}
\texttt{Buyer} \to & \texttt{Seller} & \langle \texttt{Item} \rangle\,; \\
\texttt{Seller} \to & \texttt{Buyer} & \langle \texttt{Price} \rangle\,; \\
\texttt{Buyer} \to & \texttt{Seller} & \{\texttt{OK}.\ \texttt{Seller} \to \ \texttt{Buyer}\,\langle \texttt{Deliver} \rangle\,; \\
 & & \quad\quad \texttt{Seller} \to \ \texttt{Buyer}\,\langle \texttt{Courier} \rangle\,; \mathsf{end}, \\
 & & \ \texttt{KO}.\ \mathsf{end} \\
 & & \}
\end{array}
$$

the global type associates to the two participants Buyer and Seller the types and the order of the exchanged messages

Global Type

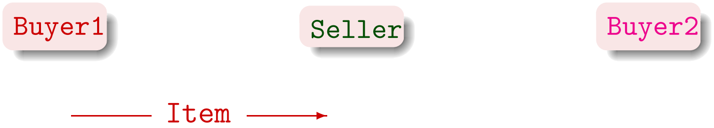
$$
\begin{array}{lll}
\texttt{Buyer} \rightarrow \texttt{Seller} & \langle\texttt{Item}\rangle\,; \\
\texttt{Seller} \rightarrow \texttt{Buyer} & \langle\texttt{Price}\rangle\,; \\
\texttt{Buyer} \rightarrow \texttt{Seller} & \{\texttt{OK}.\ \texttt{Seller} \rightarrow \texttt{Buyer}\,\langle\texttt{Deliver}\rangle\,; \\
 & \qquad \texttt{Seller} \rightarrow \texttt{Buyer}\,\langle\texttt{Courier}\rangle\,;\mathsf{end}, \\
 & \ \texttt{KO}.\ \mathsf{end} \\
 & \}
\end{array}
$$

Marco Carbone, Kohei Honda, and Nobuko Yoshida. Structured Communication-Centred Programming for Web Services. In *ESOP'07*, volume 4421 of *LNCS*, pages 2–17. Springer, 2007.

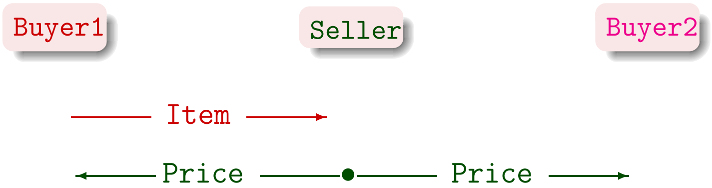
Multiparty Sessions

Buyer1 Seller Buyer2

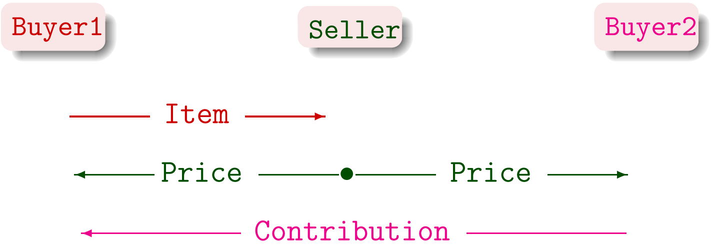
Multiparty Sessions



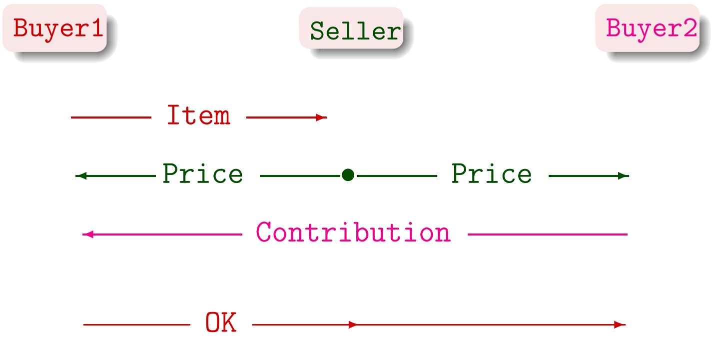
Multiparty Sessions



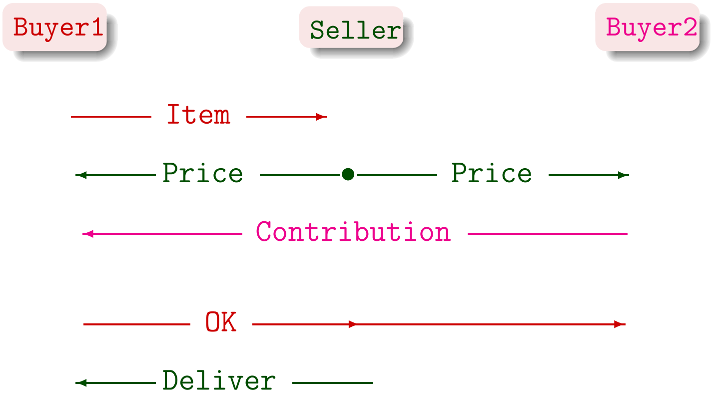
Multiparty Sessions



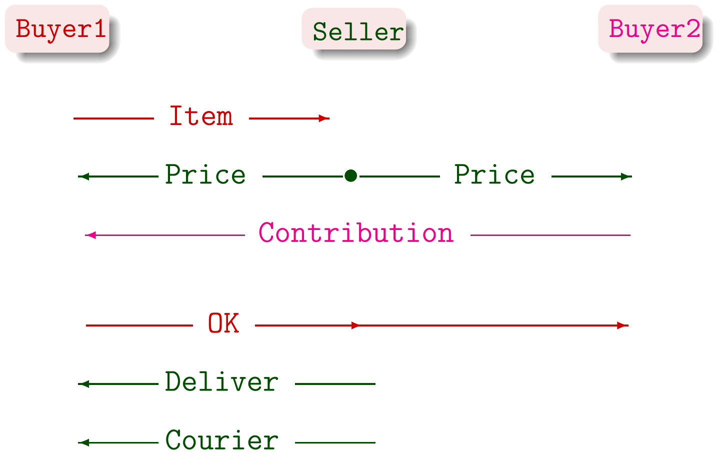
Multiparty Sessions



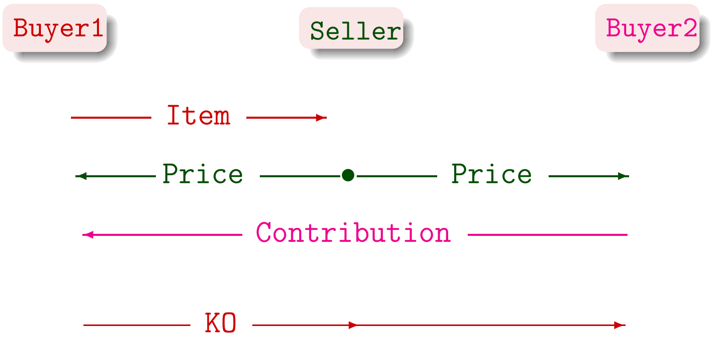
Multiparty Sessions



Multiparty Sessions



Multiparty Sessions



Global Type

$$
\begin{array}{lll}
\texttt{Buyer1} \to & \texttt{Seller} & \langle \texttt{Item} \rangle \,; \\
\texttt{Seller} \to & \begin{array}{l}\texttt{Buyer1}\\ \texttt{Buyer2}\end{array} & \langle \texttt{Price} \rangle \,; \\
\texttt{Buyer2} \to & \texttt{Buyer1} & \langle \texttt{Contr} \rangle \,; \\
\texttt{Buyer1} \to & \texttt{Seller} & \{\texttt{OK}.\ \texttt{Seller} \to \texttt{Buyer1}\, \langle \texttt{Deliver} \rangle \,; \\
 & & \qquad \texttt{Seller} \to \texttt{Buyer1}\, \langle \texttt{Courier} \rangle \,; \mathsf{end} \\
 & & \ \texttt{KO}.\ \mathsf{end} \\
 & & \}
\end{array}
$$

Global Type

$$
\begin{array}{lll}
\texttt{Buyer1} \to & \texttt{Seller} & \langle \texttt{Item} \rangle \, ; \\
\texttt{Seller} \to & \begin{array}{l}\texttt{Buyer1}\\ \texttt{Buyer2}\end{array} & \langle \texttt{Price} \rangle \, ; \\
\texttt{Buyer2} \to & \texttt{Buyer1} & \langle \texttt{Contr} \rangle \, ; \\
\texttt{Buyer1} \to & \texttt{Seller} & \{ \texttt{OK}.\ \texttt{Seller} \to \ \texttt{Buyer1} \, \langle \texttt{Deliver} \rangle \, ; \\
 & & \qquad \texttt{Seller} \to \ \texttt{Buyer1} \, \langle \texttt{Courier} \rangle \, ; \mathsf{end} \\
 & & \ \texttt{KO}.\ \mathsf{end} \\
 & & \}
\end{array}
$$

the global type associates to the participants Buyer1, Buyer2 and Seller the types and the order of the exchanged messages

Eduardo Bonelli and Adriana Compagnoni. Multipoint Session Types for a Distributed Calculus. In *TGC'07*, volume 4912 of *LNCS*, pages 240–256. Springer, 2008.

Kohei Honda, Nobuko Yoshida, and Marco Carbone. Multiparty Asynchronous Session Types. In *POPL'08*, pages 273–284. ACM, 2008.

Multirole Sessions

`Bob` : *buyer*

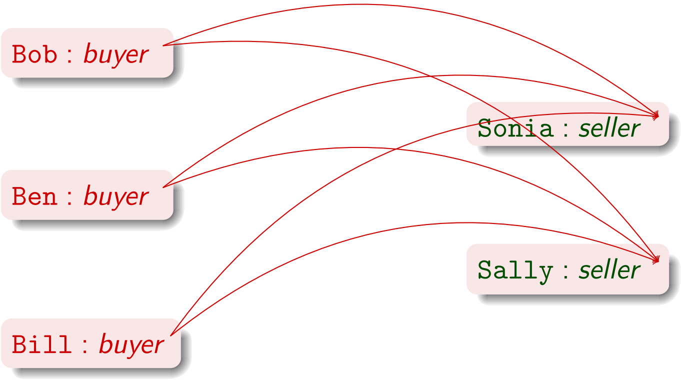
`Sonia` : *seller*

`Ben` : *buyer*

`Sally` : *seller*

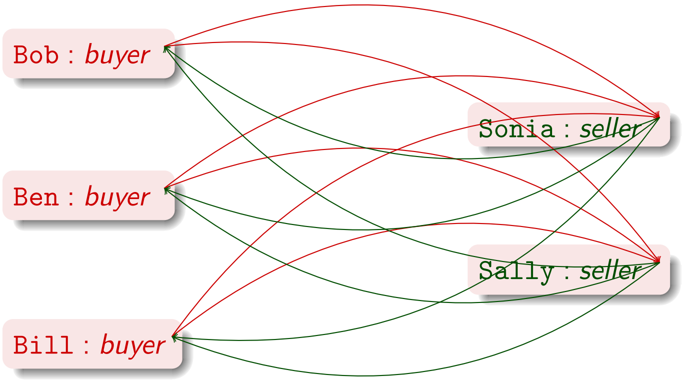
`Bill` : *buyer*

Multirole Sessions



Item

Multirole Sessions



Item

Price

Global Type

$$\forall\ \iota : \textit{buyer}.\forall\ \iota' : \textit{seller}.\quad \begin{array}{l} \iota \to \iota'\langle\texttt{Item}\rangle; \\ \iota' \to \iota\langle\texttt{Price}\rangle; \\ \iota \to \iota'\{\texttt{OK}.\iota' \to \iota\,\langle\texttt{Deliver}\rangle; \\ \qquad\qquad \iota' \to \iota\,\langle\texttt{Courier}\rangle;\mathsf{end}, \\ \qquad\quad \texttt{KO}.\mathsf{end}\} \end{array}$$

Global Type

$$\forall\ \iota : \textit{buyer}.\forall\ \iota' : \textit{seller}.\quad \begin{array}{l} \iota \to \iota'\langle\texttt{Item}\rangle; \\ \iota' \to \iota\langle\texttt{Price}\rangle; \\ \iota \to \iota'\{\texttt{OK}.\iota' \to \iota\ \langle\texttt{Deliver}\rangle; \\ \qquad\qquad \iota' \to \iota\ \langle\texttt{Courier}\rangle;\mathsf{end}, \\ \qquad \texttt{KO}.\mathsf{end}\} \end{array}$$

the global type associates to the roles *buyer* and *seller* the types and the order of the exchanged messages

Multirole Sessions

Elena Giachino, Matthew Sackman, Sophia Drossopoulou, and Susan Eisenbach. Softly safely spoken: Role playing for session types. Presented at *PLACES '09*, 2009.

Pierre-Malo Deniélou and Nobuko Yoshida. Dynamic Multirole Session Types. In *Proc. POPL'11*, pages 435–446. ACM, 2011.

Multirole Sessions with Choices

Bob : *buyer*

Sonia : *seller*

Ben : *buyer*

Sally : *seller*

Bill : *buyer*

Multirole Sessions with Choices

Bob : *buyer*

Sonia : *seller*

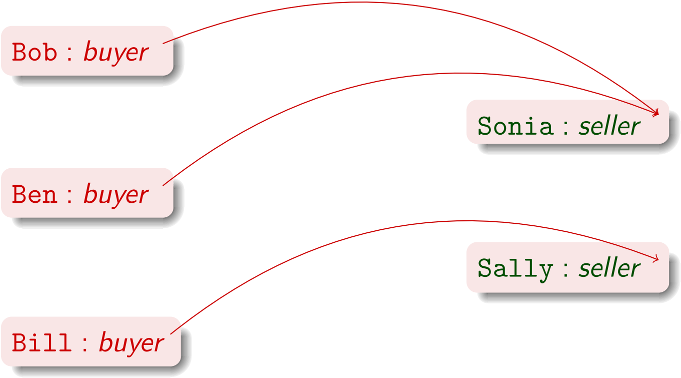
Ben : *buyer*

Sally : *seller*

Bill : *buyer*

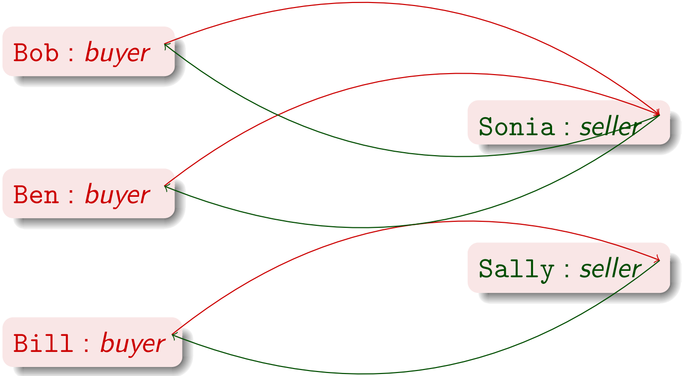
each *buyer* chooses one *seller* on the basis of *seller* reputation

Multirole Sessions with Choices



Item

Multirole Sessions with Choices



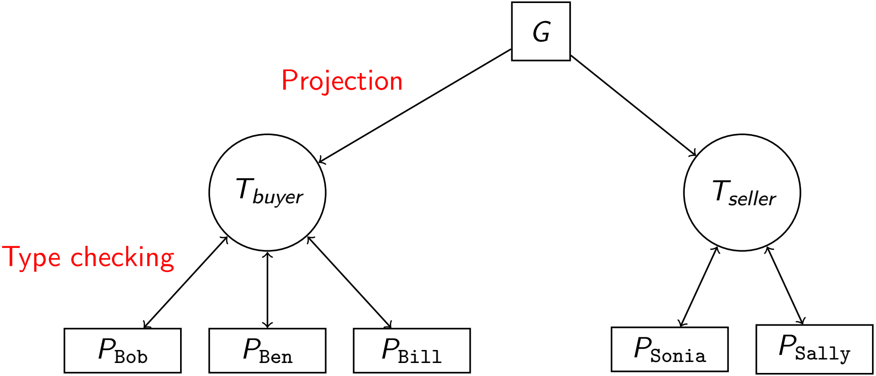
Item

Price

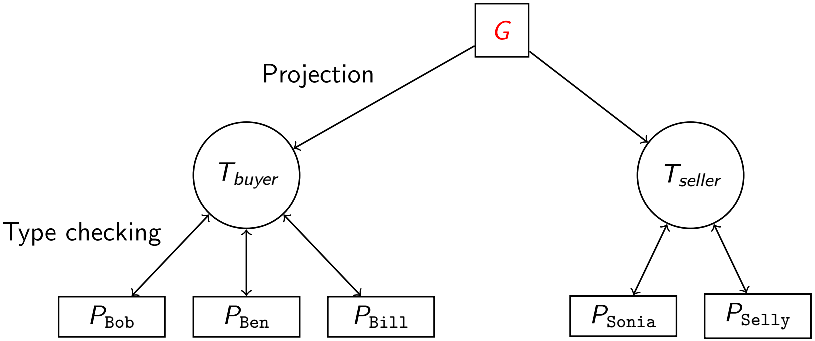
Global Type

$$\forall\ \iota : \mathit{buyer}.\ \exists\ \iota' : \mathit{seller}.\quad \begin{array}{l} \iota \to \iota'\langle\texttt{Item}\rangle; \\ \iota' \to \iota\langle\texttt{Price}\rangle; \\ \iota \to \iota'\{\texttt{OK}.\iota' \to \iota\,\langle\texttt{Deliver}\rangle; \\ \qquad\qquad \iota' \to \iota\,\langle\texttt{Courier}\rangle;\mathsf{end}, \\ \qquad\quad \texttt{KO}; \mathsf{end}\} \end{array}$$

A Three-layered Structure

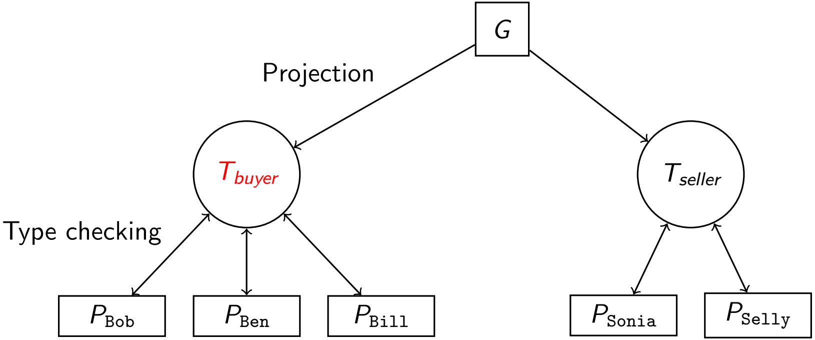


A Three-layered Structure



global type $\forall\ \iota : \mathit{buyer}.\ \exists\ \iota' : \mathit{seller}.\ \iota \to \iota'\langle \texttt{Item} \rangle; \ldots$

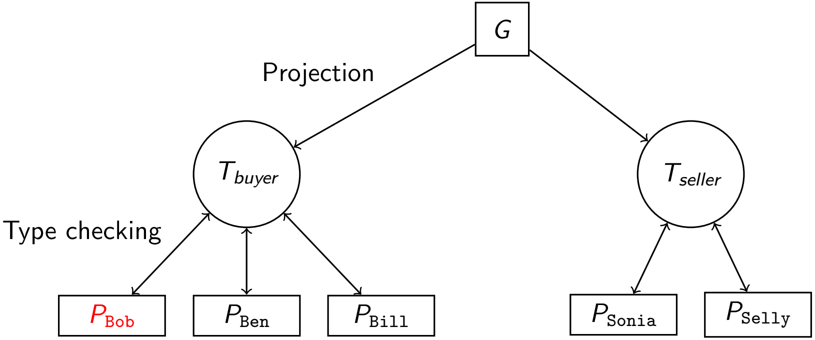
A Three-layered Structure



$\forall\ \iota : \mathit{buyer}.\ \exists\ \iota' : \mathit{seller}.\ \iota \to \iota' \langle \texttt{Item} \rangle ; \ldots$

local types $\exists\ \iota' : \mathit{seller}.!\langle \iota', \langle \texttt{Item} \rangle \rangle; \ldots$

A Three-layered Structure

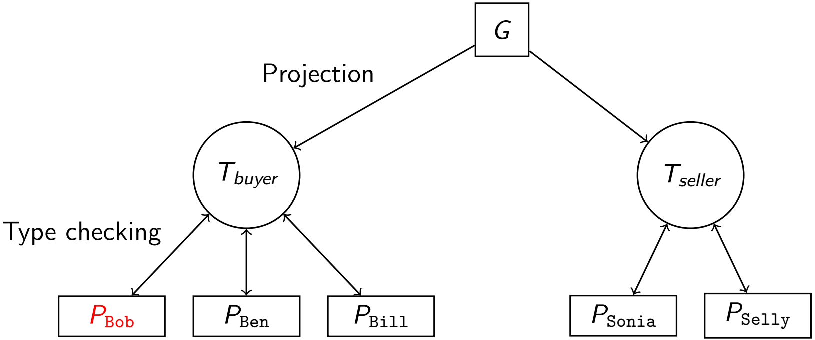


$\forall\ \iota : \mathit{buyer}.\ \exists\ \iota' : \mathit{seller}.\ \iota \to \iota'\ \langle \texttt{Item} \rangle\ ; \ldots$

$\exists\ \iota' : \mathit{seller}.!\langle \iota', \langle \texttt{Item} \rangle \rangle; \ldots$

user processes $\quad \ldots y\ \exists\ (\iota' : \mathit{seller}).\{y!\langle \iota', \langle \texttt{item} \rangle \rangle; \ldots\}$

A Three-layered Structure



$\forall\ \iota : \mathit{buyer}.\ \exists\ \iota' : \mathit{seller}.\ \iota \to \iota' \langle \texttt{Item} \rangle ; \ldots$

$\exists\ \iota' : \mathit{seller}.!\langle \iota', \langle \texttt{Item} \rangle \rangle ; \ldots$

$\ldots y\ \exists\ (\iota' : \mathit{seller}).\{y!\langle \iota', \langle \texttt{item} \rangle \rangle ; \ldots\}$

run-time processes

$\ldots s[\text{Bob} : \mathit{buyer}]\ \exists\ (\iota' : \mathit{seller}).\{s[\text{Bob} : \mathit{buyer}]!\langle \iota', \langle \texttt{item} \rangle \rangle ; \ldots\}$

Services

a service is permanent: it prescribes for a fixed set of roles

Services

a service is permanent: it prescribes for a fixed set of roles

- **the communications protocol;**

Services

a service is permanent: it prescribes for a fixed set of roles

- **the communications protocol;**

$$\forall\ \iota : \mathit{buyer}.\exists\iota' : \mathit{seller}.\quad \begin{array}{l} \iota \to \iota'\langle\texttt{Item}\rangle; \\ \iota' \to \iota\langle\texttt{Price}\rangle; \\ \iota \to \iota'\{\texttt{OK}.\iota' \to \iota\,\langle\texttt{Deliver}\rangle; \\ \qquad\qquad \iota' \to \iota\,\langle\texttt{Courier}\rangle;\mathsf{end}, \\ \qquad \texttt{KO}; \mathsf{end}\} \end{array}$$

Services

a service is permanent: it prescribes for a fixed set of roles

- **the communications protocol;**
- **the relevant interactions which will be memorised in the histories of sending principals;**

Services

a service is permanent: it prescribes for a fixed set of roles

- **the communications protocol;**
- **the relevant interactions which will be memorised in the histories of sending principals;**

$$\forall\ \iota : \mathit{buyer}.\exists \iota' : \mathit{seller}.\quad \begin{array}{l} \iota \to \iota'\langle \mathtt{Item}\rangle; \\ \iota' \to \iota\langle \mathtt{Price}\rangle; \\ \iota \to^{\bullet} \iota'\{\mathtt{OK}.\iota' \to \iota\,\langle \mathtt{Deliver}\rangle; \\ \qquad\qquad \iota' \to^{\bullet} \iota\,\langle \mathtt{Courier}\rangle;\mathsf{end}, \\ \qquad \mathtt{KO};\mathsf{end}\} \end{array}$$

the online shop will memorise:

- in the history of Bob as *buyer* the labels OK and KO;
- in the history of Sonia as *seller* the used couriers.

Services

a service is permanent: it prescribes for a fixed set of roles

- **the communications protocol;**
- **the relevant interactions which will be memorised in the histories of sending principals;**
- **the conditions on the reputations of principals which must be satisfied in order to allow them to join the service with some role.**

Services

a service is permanent: it prescribes for a fixed set of roles

- **the communications protocol;**
- **the relevant interactions which will be memorised in the histories of sending principals;**
- **the conditions on the reputations of principals which must be satisfied in order to allow them to join the service with some role.**

  `Bob` can join the online shop as *buyer* unless his last 50 transactions were unsuccessful;
  `Sonia` can join the online shop as *seller* if she used reliable couriers in the majority of transactions.

Services

a service is permanent: it prescribes for a fixed set of roles

- the communications protocol;
- the relevant interactions which will be memorised in the histories of sending principals;
- the conditions on the reputations of principals which must be satisfied in order to allow them to join the service with some role.

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**

  Bob wants to join the online shop as *buyer* only if at least one of the current sellers assures a fast delivery;

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**
- **a principal can ask to join in order to participate to all successive activations of the service (stable join) until she will decide to quit, or to participate just to the next activation of the service (one-shot join);**

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**
- **a principal can ask to join in order to participate to all successive activations of the service (stable join) until she will decide to quit, or to participate just to the next activation of the service (one-shot join);**

  `Sonia` wants to join the online shop as *seller* with a stable join;
  `Bob` wants to join the online shop as *buyer* with a one-shot join.

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**
- **a principal can ask to join in order to participate to all successive activations of the service (stable join) until she will decide to quit, or to participate just to the next activation of the service (one-shot join);**
- **a principal can join a service for one-shot only if his reputation satisfies the condition required for the role he wants to play;**

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**
- **a principal can ask to join in order to participate to all successive activations of the service (stable join) until she will decide to quit, or to participate just to the next activation of the service (one-shot join);**
- **a principal can join a service for one-shot only if his reputation satisfies the condition required for the role he wants to play;**

  Bob can join the online shop as *buyer* for one-shot unless his last 50 transactions were unsuccessful.

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**
- **a principal can ask to join in order to participate to all successive activations of the service (stable join) until she will decide to quit, or to participate just to the next activation of the service (one-shot join);**
- **a principal can join a service for one-shot only if his reputation satisfies the condition required for the role he wants to play;**
- **a principal can join a service with different roles.**

Service Join

- **a principal can ask to join a service for playing a given role under some conditions for the reputation of current participants (i.e. principals with roles);**
- **a principal can ask to join in order to participate to all successive activations of the service (stable join) until she will decide to quit, or to participate just to the next activation of the service (one-shot join);**
- **a principal can join a service for one-shot only if his reputation satisfies the condition required for the role he wants to play;**
- **a principal can join a service with different roles.**

  `Sonia` can join the online shop both as as *buyer* and as *seller*. In her history there will be labels `OK` and `KO` for her *buyer* role and the used couriers for her *seller* role.

Sessions

a session is an activation of a service

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

- **all participants who registered to the service for one-shot and**

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

- **all participants who registered to the service for one-shot and**
- **all participants who registered stably to the service and whose reputations satisfy the service conditions.**

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

- **all participants who registered to the service for one-shot and**
- **all participants who registered stably to the service and whose reputations satisfy the service conditions.**

1. If `Sonia` joined stably the online shop as *seller*, she mainly used reliable couriers and

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

- **all participants who registered to the service for one-shot and**
- **all participants who registered stably to the service and whose reputations satisfy the service conditions.**

1. If `Sonia` joined stably the online shop as *seller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

- **all participants who registered to the service for one-shot and**
- **all participants who registered stably to the service and whose reputations satisfy the service conditions.**

1. If `Sonia` joined stably the online shop as *seller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

- **all participants who registered to the service for one-shot and**
- **all participants who registered stably to the service and whose reputations satisfy the service conditions.**

1. If `Sonia` joined stably the online shop as *seller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.
4. If both `Ben` and `Bill` will join as *buyers*,

Sessions

a session is an activation of a service

many successive sessions are activated for the same service in general with different participants; more precisely when a session starts the participants are:

- **all participants who registered to the service for one-shot and**
- **all participants who registered stably to the service and whose reputations satisfy the service conditions.**

1. If `Sonia` joined stably the online shop as *seller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.
4. If both `Ben` and `Bill` will join as *buyers*,
5. then the successive session can have as participants `Ben` : *buyer*, `Bill` : *buyer* and `Sonia` : *seller*.

Sessions

Let the online shop have also the roles of

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

1. If `Sonia` joined stably the online shop both as *seller* and as *goldSeller*, she mainly used reliable couriers and

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

1. If `Sonia` joined stably the online shop both as *seller* and as *goldSeller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

1. If `Sonia` joined stably the online shop both as *seller* and as *goldSeller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

1. If `Sonia` joined stably the online shop both as *seller* and as *goldSeller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.
4. Then if `Ben` will join as *buyer*,

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

1. If `Sonia` joined stably the online shop both as *seller* and as *goldSeller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.
4. Then if `Ben` will join as *buyer*,
5. `Bill` will join as *goldBuyer* and

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

1. If `Sonia` joined stably the online shop both as *seller* and as *goldSeller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.
4. Then if `Ben` will join as *buyer*,
5. `Bill` will join as *goldBuyer* and
6. `Sonia` reached the required number of successful transactions,

Sessions

Let the online shop have also the roles of

- *goldSeller*, i.e. of sellers offering additional assistance, with the requirement of having a good record of successful transactions;
- *goldBuyer*, i.e. of buyer requiring assistance.

1. If `Sonia` joined stably the online shop both as *seller* and as *goldSeller*, she mainly used reliable couriers and
2. `Bob` joined for one-shot the online shop as *buyer*,
3. then a first session with participants `Sonia` : *seller* and `Bob` : *buyer* can start.
4. Then if `Ben` will join as *buyer*,
5. `Bill` will join as *goldBuyer* and
6. `Sonia` reached the required number of successful transactions,
7. then the successive session can have as participants `Ben` : *buyer*, `Bill` : *goldBuyer*, `Sonia` : *seller* and `Sonia` : *goldSeller*.

Poll and Choice

- **a participant can concurrently interact with all principals inhabiting a given role in a session and satisfying a given condition;**

Poll and Choice

- **a participant can concurrently interact with all principals inhabiting a given role in a session and satisfying a given condition;**

  `Sonia` : *seller* can start transactions only with buyers who did not ask for “bad” items.

Poll and Choice

- **a participant can concurrently interact with all principals inhabiting a given role in a session and satisfying a given condition;**
- **a participant can choose to interact with one of the "best" among the principals inhabiting a given role in a session and satisfying a given condition;**

Poll and Choice

- **a participant can concurrently interact with all principals inhabiting a given role in a session and satisfying a given condition;**
- **a participant can choose to interact with one of the “best” among the principals inhabiting a given role in a session and satisfying a given condition;**

  Bob : *buyer* can choose one seller with a better courier record between the sellers who are fast in delivery.

Poll and Choice

- **a participant can concurrently interact with all principals inhabiting a given role in a session and satisfying a given condition;**
- **a participant can choose to interact with one of the "best" among the principals inhabiting a given role in a session and satisfying a given condition;**

$$\forall\ \iota : \mathrm{good}(\mathit{buyer}, \iota).$$

$$\exists \iota' : \mathtt{fast}(\mathit{seller}, \iota').\quad \begin{array}{l} \iota \rightarrow^{\bullet} \iota'\langle \mathtt{Item}\rangle; \\ \iota' \rightarrow \iota\langle \mathtt{Price}\rangle; \\ \iota \rightarrow^{\bullet} \iota'\{\mathtt{OK}.\iota' \rightarrow^{\bullet} \iota\,\langle \mathtt{Deliver}\rangle; \\ \qquad\qquad \iota' \rightarrow^{\bullet} \iota\,\langle \mathtt{Courier}\rangle; \mathrm{end}, \\ \qquad \mathtt{KO}; \mathrm{end}\} \end{array}$$

Histories

Histories are exploited at various stages of the interaction:

Histories

Histories are exploited at various stages of the interaction:

- at service join, to allow the service to select the one-shot principals who will take part only to the next session and to allow a principal wishing to join the service to evaluate the reputation of the current participants and proceed or not with the join accordingly;

Histories

Histories are exploited at various stages of the interaction:

- at service join, to allow the service to select the one-shot principals who will take part only to the next session and to allow a principal wishing to join the service to evaluate the reputation of the current participants and proceed or not with the join accordingly;
- at session initiation, to allow the service to select among the stable participants those who will take part in the session, by testing if they satisfy some condition;

Histories

Histories are exploited at various stages of the interaction:

- at service join, to allow the service to select the one-shot principals who will take part only to the next session and to allow a principal wishing to join the service to evaluate the reputation of the current participants and proceed or not with the join accordingly;
- at session initiation, to allow the service to select among the stable participants those who will take part in the session, by testing if they satisfy some condition;
- in a poll operation, to allow a participant to select a bunch of other participants to interact with, according to some criteria;

Histories

Histories are exploited at various stages of the interaction:

- at service join, to allow the service to select the one-shot principals who will take part only to the next session and to allow a principal wishing to join the service to evaluate the reputation of the current participants and proceed or not with the join accordingly;
- at session initiation, to allow the service to select among the stable participants those who will take part in the session, by testing if they satisfy some condition;
- in a poll operation, to allow a participant to select a bunch of other participants to interact with, according to some criteria;
- in a choice operation, to allow a participant to select the best among the participants which satisfy some criterion.

Service Initiation

$$a\langle G, \phi\rangle$$

Service Initiation

$a\langle G, \phi\rangle$

service name

Service Initiation

$a\langle G, \phi \rangle$

service name

global type

Service Initiation

$$a\langle G, \phi\rangle$$

service name global type join condition

Service Initiation

$\mathit{onlineshop}\langle G, \mathtt{rel}\rangle$

Service Initiation

$onlineshop\langle G, \texttt{rel} \rangle$

$$G = \begin{array}{ll} \forall\ \iota : \texttt{good}(\mathit{buyer}, \iota). & \\ \exists \iota' : \texttt{fast}(\mathit{seller}, \iota'). & \iota \to^{\bullet} \iota' \langle \texttt{Item} \rangle; \\ & \iota' \to \iota \langle \texttt{Price} \rangle; \\ & \iota \to^{\bullet} \iota' \{ \texttt{OK}. \iota' \to^{\bullet} \iota \langle \texttt{Deliver} \rangle; \\ & \quad \iota' \to^{\bullet} \iota \langle \texttt{Courier} \rangle; \mathsf{end}, \\ & \quad \texttt{KO}; \mathsf{end} \} \end{array}$$

Service Initiation

$\mathit{onlineshop}\langle G, \mathtt{rel}\rangle$

$$G = \begin{array}{l} \forall\ \iota : \mathtt{good}(\mathit{buyer}, \iota). \\ \exists \iota' : \mathtt{fast}(\mathit{seller}, \iota'). \quad \begin{array}{l} \iota \to^{\bullet} \iota' \langle \mathtt{Item} \rangle; \\ \iota' \to \iota \langle \mathtt{Price} \rangle; \\ \iota \to^{\bullet} \iota' \{ \mathtt{OK}. \iota' \to^{\bullet} \iota \langle \mathtt{Deliver} \rangle; \\ \qquad\qquad \iota' \to^{\bullet} \iota \langle \mathtt{Courier} \rangle; \mathsf{end}, \\ \qquad\quad \mathtt{KO}; \mathsf{end} \} \end{array} \end{array}$$

$\mathtt{rel}(\mathit{buyer}, \mathtt{Bob})$ holds if the history of $\mathtt{Bob}$ does not end with 50 $\mathtt{KO}$
a principal can join the online shop as *buyer* unless his last 50 transactions were unsuccessful

$\mathit{onlineshop}\langle G, \mathtt{rel}\rangle$

$$G = \begin{array}{ll} \forall\ \iota : \mathtt{good}(\mathit{buyer}, \iota). & \\ \exists \iota' : \mathtt{fast}(\mathit{seller}, \iota'). & \iota \to^{\bullet} \iota'\langle \mathtt{Item}\rangle; \\ & \iota' \to \iota\langle \mathtt{Price}\rangle; \\ & \iota \to^{\bullet} \iota'\{\mathtt{OK}.\iota' \to^{\bullet} \iota\,\langle \mathtt{Deliver}\rangle; \\ & \qquad \iota' \to^{\bullet} \iota\,\langle \mathtt{Courier}\rangle; \mathsf{end}, \\ & \quad \mathtt{KO}; \mathsf{end}\} \end{array}$$

$\mathtt{rel}(\mathit{buyer}, \mathtt{Bob})$ holds if the history of Bob does not end with 50 KO
a principal can join the online shop as *buyer* unless his last 50 transactions were unsuccessful

$\mathtt{rel}(\mathit{seller}, \mathtt{Sonia})$ holds if at least half of the couriers in the history of Sonia are reliable
a principal can join the online shop as *seller* if she used reliable couriers in the majority of transactions

Service Initiation

$$a\langle G, \phi\rangle \longrightarrow a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

Service Initiation

$$a\langle G, \phi\rangle \longrightarrow a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

service registry

Service Initiation

$$a\langle G, \phi\rangle \longrightarrow a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

service registry

$$a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

Service Initiation

$$a\langle G, \phi\rangle \longrightarrow a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

service registry

$$a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

set of histories

Service Initiation

$$a\langle G, \phi\rangle \longrightarrow a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

service registry

$$a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

set of histories stable participants

Service Initiation

$$a\langle G, \phi\rangle \longrightarrow a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

service registry

$$a[\emptyset, \mathbf{0}, \mathbf{0}, \phi]$$

set of histories   stable participants   one-shot participants

Service Initiation

$$\mathit{onlineshop}\langle G, \mathtt{rel}\rangle \longrightarrow \mathit{onlineshop}[\emptyset, \mathbf{0}, \mathbf{0}, \mathtt{rel}]$$

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$$

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$$

principal

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$$

principal role

Stable join

$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$

principal role conditions on current participants

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$$

principal   role   conditions on current participants   private channel

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$$

$$\overline{onlineshop}[\mathtt{Sonia} : seller, \exists\iota.\mathtt{good}(\mathtt{buyer}, \iota)](y_s).P_s$$

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1 \mid [\mathtt{id} : r](y).P, Q_2, \phi]$$

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1 \mid [\mathtt{id} : r](y).P, Q_2, \phi]$$

`id` is added with the empty history to $\mathscr{H}$ only if she does not already have an history in $\mathscr{H}$

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1 \mid [\mathtt{id} : r](y).P, Q_2, \phi]$$

$[\mathtt{id} : r](y).P$ is put in parallel to Q_1

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1 \mid [\mathtt{id} : r](y).P, Q_2, \phi]$$

if $(\mathtt{id} : r) \notin Q_1 \mid Q_2$ and $\mathscr{C}(\tilde{r})\mathscr{H}$

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1 \mid [\mathtt{id} : r](y).P, Q_2, \phi]$$

if $(\mathtt{id} : r) \notin Q_1 \mid Q_2$ and $\mathscr{C}(\tilde{r})\mathscr{H}$

the participant is not already present

Stable join

$$\overline{a}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1 \mid [\mathtt{id} : r](y).P, Q_2, \phi]$$

if $(\mathtt{id} : r) \notin Q_1 \mid Q_2$ and $\mathscr{C}(\tilde{r})\mathscr{H}$

the current participants satisfy the conditions $\mathscr{C}(\tilde{r})$

Stable join

$$\underline{\mathit{onlineshop}}[\mathscr{H}, \mathbf{0}, \mathbf{0}, \texttt{rel}] \mid \overline{\mathit{onlineshop}}[\texttt{Sonia} : \mathit{seller}, \exists\iota.\texttt{good}(\texttt{buyer}, \iota)](y_s).P_s \longrightarrow$$

$$\mathit{onlineshop}[\mathscr{H} \cup \{(\texttt{Sonia}, \circ)\}, [\texttt{Sonia} : \mathit{seller}](y_s).P_s, \mathbf{0}, \texttt{rel}]$$

One-shot Join

$$a[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$$

One-shot Join

$$a[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P$$

$$\mathit{onlineshop}[\mathtt{Bob} : \mathit{buyer}, \exists\iota.\mathtt{fast}(\mathtt{seller}, \iota)](y_b).P_b$$

One-shot Join

$$a[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1, Q_2 \mid [\mathtt{id} : r](y).P, \phi]$$

One-shot Join

$$a[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1, Q_2 \mid [\mathtt{id} : r](y).P, \phi]$$

if $\phi(\mathscr{H} \restriction \mathtt{id}, r)$ and $(\mathtt{id} : r) \notin Q_1 \mid Q_2$ and $\mathscr{C}(\tilde{r})\mathscr{H}$

One-shot Join

$$a[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \mid a[\mathscr{H}, Q_1, Q_2, \phi] \longrightarrow$$
$$a[\mathscr{H} \flat \mathtt{id}, Q_1, Q_2 \mid [\mathtt{id} : r](y).P, \phi]$$

if $\phi(\mathscr{H} \lceil \mathtt{id}, r)$ and $(\mathtt{id} : r) \notin Q_1 \mid Q_2$ and $\mathscr{C}(\tilde{r})\mathscr{H}$

the history of id satisfies ϕ for role r

One-shot Join

$$\mathit{onlineshop}[\{(\mathtt{Sonia}, \circ)\}, [\mathtt{Sonia} : \mathit{seller}](y_s).P_s, \mathbf{0}, \mathtt{rel}] \mid \mathit{onlineshop}[\mathtt{Bob} : \mathit{buyer}, \exists\iota.\mathtt{fast}(\mathtt{seller}, \iota)](y_b).P_b \longrightarrow$$

$$\mathit{onlineshop}[\{(\mathtt{Bob}, \circ), (\mathtt{Sonia}, \circ)\}, [\mathtt{Sonia} : \mathit{seller}](y_s).P_s, [\mathtt{Bob} : \mathit{buyer}](y_b).P_b, \mathtt{rel}]$$

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi]$$

$\forall i \in I.\phi(\mathscr{H}\upharpoonright\mathtt{id}_i, r_i)$ and $\forall(\mathtt{id} : r) \in Q.\neg\phi(\mathscr{H}\upharpoonright\mathtt{id}, r)$:
in the service register we distinguish the stable participants according to those who satisfy or not the join condition ϕ

Session Initiation

$$a[\mathscr{H}, \Pi_{i \in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j \in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

fresh and private session name s

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

session registry with the set of participants

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

$$\Pi_{i\in I\cup J} P_i\{s[\mathtt{id}_i : r_i]/y_i\} \mid s : [\,]) \mid$$

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

$$\Pi_{i\in I\cup J} P_i\{s[\mathtt{id}_i : r_i]/y_i\} \mid s : [\,]) \mid$$

session participants where the private channels have been replaced by run time channels (session name [principal identity: role])

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

$$\Pi_{i\in I\cup J} P_i\{s[\mathtt{id}_i : r_i]/y_i\} \mid s : [\,]) \mid$$

empty message buffer

Session Initiation

$$a[\mathscr{H}, \Pi_{i \in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j \in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

$$\Pi_{i \in I \cup J} P_i\{s[\mathtt{id}_i : r_i]/y_i\} \mid s : [\,]) \mid$$

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

$$\Pi_{i\in I\cup J} P_i\{s[\mathtt{id}_i : r_i]/y_i\} \mid s : [\,]) \mid$$

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \mathbf{0}, \phi]$$

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

$$\Pi_{i\in I\cup J} P_i\{s[\mathtt{id}_i : r_i]/y_i\} \mid s : [\,]) \mid$$

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \mathbf{0}, \phi]$$

the same stable participants and no one-shot participant

Session Initiation

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \Pi_{j\in J}[\mathtt{id}_j : r_j](y_j).P_j, \phi] \longrightarrow$$

$$(\nu s)(a\langle s, \{\mathtt{id}_i : r_i | i \in I \cup J\}\rangle \mid$$

$$\Pi_{i\in I\cup J} P_i\{s[\mathtt{id}_i : r_i]/y_i\} \mid s : [\,]) \mid$$

$$a[\mathscr{H}, \Pi_{i\in I}[\mathtt{id}_i : r_i](y_i).P_i \mid Q, \mathbf{0}, \phi]$$

Session Initiation

$$onlineshop[-, [\texttt{Sonia} : seller](y_s).P_s, [\texttt{Bob} : buyer](y_b).P_b, -] \longrightarrow$$

$$(\nu s)(onlineshop\langle s, \{\texttt{Sonia} : seller, \texttt{Bob} : buyer\}\rangle \mid$$

$$P_s\{s[\texttt{Sonia} : seller]/y_s\} \mid P_b\{s[\texttt{Bob} : buyer]/y_b\} \mid s{:}[\,]) \mid$$

$$onlineshop[-, [\texttt{Sonia} : seller](y_s).P_s, \mathbf{0}, -]$$

Asynchronous Communication

in session s

$$s[\mathtt{id} : r]!\langle \mathtt{id}' : r', l\langle v\rangle\rangle$$

participant $\mathtt{id}$:r sends label l and value v to participant $\mathtt{id}' : r'$

Asynchronous Communication

in session s

participant $\mathtt{id}{:}r$ sends label l and value v to participant $\mathtt{id}' : r'$

$$s[\mathtt{id} : r]!\langle \mathtt{id}' : r', l\langle v\rangle\rangle \mid s : \mathscr{B} \longrightarrow s : \mathscr{B} \cdot (\mathtt{id} : r, \mathtt{id}' : r', l\langle v\rangle)$$

the message $(\mathtt{id} : r, \mathtt{id}' : r', l\langle v\rangle)$ is created and added to the buffer

Asynchronous Communication

in session s

$$s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle$$

participant $\mathtt{id}$:r waits for a label l_i with $i \in I$ and value to replace α_i from participant $\mathtt{id}' : r'$

Asynchronous Communication

in session s

participant $\mathtt{id}{:}r$ waits for a label l_i with $i \in I$ and value to replace α_i from participant $\mathtt{id}' : r'$

$$s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle \mid s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B} \longrightarrow P_k\{v/\alpha_k\} \mid s : \mathscr{B}$$

where $k \in I$

the message $(\mathtt{id} : r, \mathtt{id}' : r', l\langle v\rangle)$ is taken from the buffer

Histories

Histories are built by recording some of the labels and values that are sent by principals together with the roles in which the principals are

Histories

Histories are built by recording some of the labels and values that are sent by principals together with the roles in which the principals are

in session s

$$s[\mathtt{id} : r]!^{\bullet}\langle \mathtt{id}' : r', l\langle v\rangle\rangle$$

participant $\mathtt{id}$:r sends label l and a value v to participant $\mathtt{id}' : r'$
the message content $l\langle v\rangle$ and the role r are recorded in the history of principal $\mathtt{id}$

Histories

Histories are built by recording some of the labels and values that are sent by principals together with the roles in which the principals are

in session s

participant $\texttt{id}{:}r$ sends label l and a value v to participant $\texttt{id}' : r'$
the message content $l\langle v\rangle$ and the role r are recorded in the history of principal $\texttt{id}$

$$s[\texttt{id} : r]!^{\bullet}\langle \texttt{id}' : r', l\langle v\rangle\rangle \mid a[\mathscr{H} \cup (\texttt{id}, ⓗ), Q_1, Q_2, \phi] \mid s : \mathscr{B} \longrightarrow$$

$$a[\mathscr{H} \cup (\texttt{id}, ⓗ \cdot (l\langle v\rangle, r)), Q_1, Q_2, \phi] \mid s : \mathscr{B} \cdot (\texttt{id} : r, \texttt{id}' : r', l\langle v\rangle)$$

Histories

Histories are built by recording some of the labels and values that are sent by principals together with the roles in which the principals are

in session s

$$s[\mathtt{Bob} : \mathit{buyer}]!^{\bullet}\langle \mathtt{Sonia} : \mathit{seller}, \mathtt{OK}\rangle \mid$$
$$\mathit{onlineshop}[(\mathtt{Bob}, \circ), (\mathtt{Sonia}, \circ), -, -, -] \mid s : [\,] \longrightarrow$$
$$\mathit{onlineshop}[(\mathtt{Bob}, (\mathtt{OK}, \mathit{buyer})), (\mathtt{Sonia}, \circ), -, -, -] \mid$$
$$s : (\mathtt{Bob} : \mathit{buyer}, \mathtt{Sonia} : \mathit{seller}, \mathtt{OK})$$

Poll

in session s

$$s[\mathtt{id} : r]\forall(\iota : \mathtt{C}(r', \iota)).\{P\}$$

participant $\mathtt{id}{:}r$ want to interact (following P) with all other principals playing role r' in session s whose histories satisfy $\mathtt{C}(r', \iota)$

Poll

in session s

participant id:r want to interact (following P) with all other principals playing role r' in session s whose histories satisfy $\mathtt{C}(r', \iota)$

$$s[\mathtt{id} : r]\forall(\iota : \mathtt{C}(r', \iota)).\{P\}\longrightarrow$$

$$\Pi_{i \in I} s[\mathtt{id} : r]!\langle \mathtt{id}_i : r', \mathtt{YES}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}_i : r', \{\mathtt{YES}.P\{\mathtt{id}_i/\iota\}, \mathtt{NO}.\mathbf{0}\}\rangle$$

participant id:r sends YES to all the participants she want to interact with and waits for their answers

Poll

in session s

participant $\mathtt{id}{:}r$ want to interact (following P) with all other principals playing role r' in session s whose histories satisfy $\mathtt{C}(r',\iota)$

$$s[\mathtt{id} : r]\forall(\iota : \mathtt{C}(r',\iota)).\{P\}\longrightarrow$$

$$\Pi_{i\in I} s[\mathtt{id} : r]!\langle \mathtt{id}_i : r', \mathtt{YES}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}_i : r', \{\mathtt{YES}.P\{\mathtt{id}_i/\iota\}, \mathtt{NO}.\mathbf{0}\}\rangle \mid$$
$$\Pi_{j\in J} s[\mathtt{id} : r]!\langle \mathtt{id}_j : r', \mathtt{NO}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}_j : r', \{\mathtt{YES}.\mathbf{0}, \mathtt{NO}.\mathbf{0}\}\rangle$$

participant $\mathtt{id}{:}r$ sends also NO to all the participants she does not want to interact with and waits for their answers

Poll

in session s

participant $\mathtt{id}$:r want to interact (following P) with all other principals playing role r' in session s whose histories satisfy $\mathtt{C}(r', \iota)$

$$s[\mathtt{id} : r]\forall(\iota : \mathtt{C}(r', \iota)).\{P\}\longrightarrow$$

$$\Pi_{i\in I} s[\mathtt{id} : r]!\langle \mathtt{id}_i : r', \mathtt{YES}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}_i : r', \{\mathtt{YES}.P\{\mathtt{id}_i/\iota\}, \mathtt{NO}.\mathbf{0}\}\rangle \mid$$
$$\Pi_{j\in J} s[\mathtt{id} : r]!\langle \mathtt{id}_j : r', \mathtt{NO}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}_j : r', \{\mathtt{YES}.\mathbf{0}, \mathtt{NO}.\mathbf{0}\}\rangle$$

where I and J are computed using the set $\mathscr{P}$ of participants to session s and the history $\mathscr{H}$ memorised in the current service:

$$\{\mathtt{id}_i \mid i \in I\} = \{\mathtt{id}' \neq \mathtt{id} \mid (\mathtt{id}' : r') \in \mathscr{P} \ \&\ \mathtt{C}(r', \mathtt{id}')\mathscr{H}\}$$
$$\{\mathtt{id}_j \mid j \in J\} = \{\mathtt{id}' \neq \mathtt{id} \mid (\mathtt{id}' : r') \in \mathscr{P} \ \&\ \neg\mathtt{C}(r', \mathtt{id}')\mathscr{H}\}$$

Poll

in session s

$$s[\texttt{Sonia} : \mathit{seller}]\forall(\iota : \mathsf{good}(\mathit{buyer}, \iota)). \quad \longrightarrow$$
$$\{s[\texttt{Sonia} : \mathit{seller}]?\langle \iota, (x). \ldots \rangle\}$$

$$\begin{array}{l}
s[\texttt{Sonia} : \mathit{seller}]!\langle \texttt{Bob} : \mathit{buyer}, \texttt{YES}\rangle; \\
s[\texttt{Sonia} : \mathit{seller}]?\langle \texttt{Bob} : \mathit{buyer}, \{\texttt{YES}.s[\texttt{Sonia} : \mathit{seller}]?\langle \texttt{Bob} : \mathit{buyer}, (x). \ldots \rangle, \\
\qquad\qquad\qquad\qquad \texttt{NO}.\mathbf{0}\} \mid \\
s[\texttt{Sonia} : \mathit{seller}]!\langle \texttt{Ben} : \mathit{buyer}, \texttt{YES}\rangle; \\
s[\texttt{Sonia} : \mathit{seller}]?\langle \texttt{Ben} : \mathit{buyer}, \{\texttt{YES}.s[\texttt{Sonia} : \mathit{seller}]?\langle \texttt{Ben} : \mathit{buyer}, (x). \ldots \rangle, \\
\qquad\qquad\qquad\qquad \texttt{NO}.\mathbf{0}\} \mid \\
s[\texttt{Sonia} : \mathit{seller}]!\langle \texttt{Bill} : \mathit{buyer}, \texttt{NO}\rangle; \\
s[\texttt{Sonia} : \mathit{seller}]?\langle \texttt{Bill} : \mathit{buyer}, \{\texttt{YES}.\mathbf{0} \\
\qquad\qquad\qquad\qquad \texttt{NO}.\mathbf{0}\}
\end{array}$$

Choice

in session s

$$s[\mathtt{id} : r]\exists(\iota : \mathtt{C}(r', \iota)).\{P\}$$

participant id:r want to interact (following P) with one of the best principals playing role r' in session s whose histories satisfy $\mathtt{C}(r', \iota)$

Choice

in session s

participant id:r want to interact (following P) with one of the best principals playing role r' in session s whose histories satisfy $\mathtt{C}(r', \iota)$

$$s[\mathtt{id} : r]\exists(\iota : \mathtt{C}(r', \iota)).\{P\} \longrightarrow$$

$$s[\mathtt{id} : r]!\langle \mathtt{id}' : r', \mathtt{YES}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{\mathtt{YES}.P\{\mathtt{id}'/\iota\}, \mathtt{NO}.\mathbf{0}\}\rangle$$

participant id:r sends YES to the choosed participant and waits for her answer

Choice

in session s

participant $\mathtt{id}$:r want to interact (following P) with one of the best principals playing role r' in session s whose histories satisfy $\mathtt{C}(r',\iota)$

$$s[\mathtt{id}:r]\exists(\iota:\mathtt{C}(r',\iota)).\{P\}\longrightarrow$$

$$s[\mathtt{id}:r]!\langle\mathtt{id}':r',\mathtt{YES}\rangle;s[\mathtt{id}:r]?\langle\mathtt{id}':r',\{\mathtt{YES}.P\{\mathtt{id}'/\iota\},\mathtt{NO}.\mathbf{0}\}\rangle\mid$$
$$\Pi_{j\in J}s[\mathtt{id}:r]!\langle\mathtt{id}_j:r',\mathtt{NO}\rangle;s[\mathtt{id}:r]?\langle\mathtt{id}_j:r',\{\mathtt{YES}.\mathbf{0},\mathtt{NO}.\mathbf{0}\}\rangle$$

participant $\mathtt{id}$:r sends also $\mathtt{NO}$ to all the other participant in role r' and waits for their answers

Choice

in session s

participant $\mathtt{id}$:r want to interact (following P) with one of the best principals playing role r' in session s whose histories satisfy $\mathtt{C}(r', \iota)$

$$s[\mathtt{id} : r]\exists(\iota : \mathtt{C}(r', \iota)).\{P\} \longrightarrow$$

$$s[\mathtt{id} : r]!\langle \mathtt{id}' : r', \mathtt{YES}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{\mathtt{YES}.P\{\mathtt{id}'/\iota\}, \mathtt{NO}.\mathbf{0}\}\rangle \mid \\ \Pi_{j \in J} s[\mathtt{id} : r]!\langle \mathtt{id}_j : r', \mathtt{NO}\rangle; s[\mathtt{id} : r]?\langle \mathtt{id}_j : r', \{\mathtt{YES}.\mathbf{0}, \mathtt{NO}.\mathbf{0}\}\rangle$$

where $\mathtt{id}'$ and J are computed using the set $\mathscr{P}$ of participants to session s, the history $\mathscr{H}$ memorised in the current service and a a partial order between reputations for role r' ($\sqsubseteq_{r'}$):
$(\mathtt{id}' : r') \in \mathscr{P}$ and $\mathtt{C}(r', \mathtt{id}')\mathscr{H}$ and

$$\not\exists \mathtt{id}'' \neq \mathtt{id}.(\mathtt{id}'' : r') \in \mathscr{P} \;\&\; \mathtt{C}(r', \mathtt{id}'')\mathscr{H} \Rightarrow \mathscr{H}\lceil \mathtt{id}' \sqsubseteq_{r'} \mathscr{H}\lceil \mathtt{id}''$$

$\{\mathtt{id}_j \mid j \in J\} = \{\mathtt{id}'' \neq \mathtt{id} \mid (\mathtt{id}'' : r') \in \mathscr{P} \;\&\; \mathtt{id}'' \neq \mathtt{id}'\}$

Choice

in session s

$$s[\texttt{Bob} : \mathit{buyer}]\exists(\iota : \texttt{fast}(\mathit{seller}, \iota)). \{s[\texttt{Bob} : \mathit{buyer}]!\langle\iota, \langle\texttt{item}\rangle\rangle; \ldots\} \longrightarrow$$

$$s[\texttt{Bob} : \mathit{buyer}]!\langle\texttt{Sonia} : \mathit{seller}, \texttt{YES}\rangle;$$
$$s[\texttt{Bob} : \mathit{buyer}]?\langle\texttt{Sonia} : \mathit{seller}, \{\texttt{YES}. \; s[\texttt{Bob} : \mathit{buyer}]?\langle\texttt{Sonia} : \mathit{seller}, \langle\texttt{item}\rangle\rangle; \ldots \; \texttt{NO}.\mathbf{0}\} \mid$$
$$s[\texttt{Bob} : \mathit{buyer}]!\langle\texttt{Sally} : \mathit{seller}, \texttt{NO}\rangle;$$
$$s[\texttt{Bob} : \mathit{buyer}]?\langle\texttt{Sally} : \mathit{seller}, \{\texttt{YES}.\mathbf{0} \; \texttt{NO}.\mathbf{0}\}$$

Type System

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

Type System

$$\Gamma \vdash P \triangleright \color{red}{\Delta}$$

session environment associates local types with channel variables

$$\Delta ::= \emptyset \mid c : T$$

Type System

$$\color{red}{\Gamma} \vdash P \triangleright \Delta$$

standard environment associates sorts with value variables, global sorts with service names, roles with principal variables, and session environments with process variables

$$\Gamma ::= \emptyset \parallel \Gamma, x : S \parallel a : \langle G \rangle \parallel \Gamma, \iota : r \parallel \Gamma, X : \Delta$$

Type System

the actions of different principals can only be sequentialised via some communication

$$\Gamma \vdash_{\texttt{id}} P \triangleright \Delta$$

Type System

the actions of different principals can only be sequentialised via some communication

$$\frac{\Gamma \vdash u : \langle G \rangle \quad \Gamma \vdash_{\mathtt{id}} P \triangleright \Delta, y : G \upharpoonright (\mathtt{id} : r)}{\Gamma \vdash_{\mathtt{id}} \overline{u}[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \triangleright \Delta}$$

Type System

the actions of different principals can only be sequentialised via some communication

$$\frac{\Gamma \vdash u : \langle G \rangle \quad \Gamma \vdash_{\mathtt{id}} P \triangleright \Delta, y : G \upharpoonright (\mathtt{id} : r)}{\Gamma \vdash_{\mathtt{id}} u[\mathtt{id} : r, \mathscr{C}(\tilde{r})](y).P \triangleright \Delta}$$

Type System

the actions of different principals can only be sequentialised via some communication

$$\frac{\Gamma \vdash_{\texttt{id}} P_1 \rhd \Delta_1 \quad \Gamma \vdash_{\texttt{id}} P_2 \rhd \Delta_2}{\Gamma \vdash_{\texttt{id}} P_1; P_2 \rhd \Delta_1; \Delta_2}$$

Online Shop

$$\forall\ \iota : \mathsf{good}(buyer, \iota).\ \exists \iota : \mathtt{fast}(seller, \iota').\ G$$

$$\forall\ \iota : goldBuyer.\ \exists(\iota' : \mathtt{fast}(goldSeller, \iota')).\ \iota \to^{\bullet} \iota'\{\mathtt{BUY}.G,\ \mathtt{AST}.\iota \to \iota'\langle\mathtt{Help}\rangle; \iota' \to \iota\langle\mathtt{Solution}\rangle;\mathsf{end}\}$$

$$G = \quad \iota \to^{\bullet} \iota'\langle\mathtt{Item}\rangle;\ \iota' \to \iota\langle\mathtt{Price}\rangle;\ \iota \to^{\bullet} \iota'\{\mathtt{OK}.\iota' \to^{\bullet} \iota\,\langle\mathtt{Deliver}\rangle;\ \iota' \to^{\bullet} \iota\,\langle\mathtt{Courier}\rangle;\mathsf{end},\ \mathtt{KO}; \mathsf{end}\}$$

Online Shop

$$
\begin{array}{l}
\mathit{onlineshop}[\texttt{Bob} : \mathit{buyer}, \texttt{true}](y). \\
\quad y\exists(\iota : \texttt{fast}(\mathit{seller}, \iota)).\ \{y!^{\bullet}\langle \iota, \langle \texttt{item}\rangle\rangle; \\
\quad y?\langle \iota, (x). \\
\quad \texttt{if}\ \ \texttt{OK}(x)\ \texttt{then}\ y!^{\bullet}\langle \iota, \texttt{OK}\rangle; y?\langle \iota, (x_1).y?\langle \iota, (x_2).\mathbf{0}\rangle\rangle \\
\qquad\qquad\quad \texttt{else}\ \ y!^{\bullet}\langle \iota, \texttt{KO}\rangle; \mathbf{0}\rangle\}
\end{array}
$$

$$
\begin{array}{l}
\overline{\mathit{onlineshop}}[\texttt{Sonia} : \mathit{seller}, \texttt{true}](y). \\
\quad y\forall(\iota : \texttt{good}(\mathit{buyer}, \iota)).\{y?\langle \iota, (x).y!\langle \iota, \texttt{price}\rangle; \\
\quad y?\langle \iota, \{\texttt{OK}.y!^{\bullet}\langle \iota, \texttt{deliver}\rangle; y!^{\bullet}\langle \iota, \texttt{courier}\rangle; \mathbf{0}, \\
\qquad\quad \texttt{KO}.\mathbf{0}\}\rangle\rangle\}
\end{array}
$$

Online Shop

$$
\begin{array}{l}
\mathit{onlineshop}[\texttt{Ben} : \mathit{goldBuyer}, \exists\iota.\texttt{fast}(\mathit{seller}, \iota)](y).\\
\quad y\exists(\iota : \texttt{fast}(\mathit{goldSeller})).\\
\quad \{\ \texttt{if}\ \ \texttt{buy then}\ y!^{\bullet}\langle\iota, \texttt{BUY}\rangle; y!^{\bullet}\langle\iota, \texttt{item}\rangle; y?\langle\iota, (x).\\
\qquad\qquad \texttt{if}\ \ \texttt{OK}(x)\ \texttt{then}\ y!^{\bullet}\langle\iota, \texttt{OK}\rangle; y?\langle\iota, (x_1).y?\langle\iota, (x_2).\mathbf{0}\rangle\rangle\\
\qquad\qquad\qquad\qquad \texttt{else}\ \ y!^{\bullet}\langle\iota, \texttt{KO}\rangle; \mathbf{0}\rangle\\
\qquad\qquad \texttt{else}\ y!^{\bullet}\langle\iota, \texttt{AST}\rangle; y!\langle\iota, \texttt{help}\rangle; y?\langle\iota, (x').\mathbf{0}\rangle\}
\end{array}
$$

$$
\begin{array}{l}
\overline{\mathit{onlineshop}}[\texttt{Sonia} : \mathit{goldSeller}, \exists\iota.\texttt{good}(\mathit{buyer}, \iota)](y).\\
\quad y\forall(\iota : \mathit{goldBuyer}).\\
\quad \{y?\langle\iota, \{\texttt{BUY}.\ \ y?\langle\iota, (x).y!\langle\iota, \texttt{price}\rangle;\\
\qquad\quad y?\langle\iota, \{\texttt{OK}.y!^{\bullet}\langle\iota, \texttt{deliver}\rangle; y!^{\bullet}\langle\iota, \texttt{courier}\rangle; \mathbf{0},\\
\qquad\qquad\quad \texttt{KO}.\mathbf{0}\}\rangle\rangle\\
\qquad\quad \texttt{AST}.y?\langle\iota, (x').y!^{\bullet}\langle\iota, \texttt{solution}\rangle; \mathbf{0}\rangle\}\rangle\}
\end{array}
$$

Subject Reduction

reduction of session environments:

Subject Reduction

reduction of session environments:

$$\{s[\mathtt{id} : r] : \forall\ \iota : \mathtt{C}(r', \iota).T\} \Rightarrow$$
$$\{s[\mathtt{id} : r] : \Pi_{i \in I} !\langle \mathtt{id}_i : r', \mathtt{YES}\rangle; ?\langle \mathtt{id}_i : r', \{\mathtt{YES}.T\{\mathtt{id}_i/\iota\}, \mathtt{NO}.\epsilon\}\rangle \mid$$
$$\Pi_{j \in J} !\langle \mathtt{id}_j : r', \mathtt{NO}\rangle; ?\langle \mathtt{id}_j : r', \{\mathtt{YES}.\epsilon, \mathtt{NO}.\epsilon\}\}$$

Subject Reduction

reduction of session environments:

$$\{s[\mathtt{id} : r] : \forall\ \iota : \mathtt{C}(r', \iota).T\} \Rightarrow$$
$$\{s[\mathtt{id} : r] : \Pi_{i \in I}!\langle \mathtt{id}_i : r', \mathtt{YES}\rangle; ?\langle \mathtt{id}_i : r', \{\mathtt{YES}.T\{\mathtt{id}_i/\iota\}, \mathtt{NO}.\epsilon\}\rangle \mid$$
$$\Pi_{j \in J}!\langle \mathtt{id}_j : r', \mathtt{NO}\rangle; ?\langle \mathtt{id}_j : r', \{\mathtt{YES}.\epsilon, \mathtt{NO}.\epsilon\}\}$$

$$\{s[\mathtt{id} : r] : \exists\ \iota : \mathtt{C}(r', \iota).T\} \Rightarrow$$
$$\{s[\mathtt{id} : r] : !\langle \mathtt{id}' : r', \mathtt{YES}\rangle; ?\langle \mathtt{id}' : r', \{\mathtt{YES}.T\{\mathtt{id}'/\iota\}, \mathtt{NO}.\epsilon\}\rangle \mid$$
$$\Pi_{j \in J}!\langle \mathtt{id}_j : r', \mathtt{NO}\rangle; ?\langle \mathtt{id}_j : r', \{\mathtt{YES}.\epsilon, \mathtt{NO}.\epsilon\}\}$$

Subject Reduction

a process is initial if it does not contain free variables and runtime syntax

Subject Reduction

a process is *initial* if it does not contain free variables and runtime syntax

a process P is reachable if there is a well-typed initial process P_0 such that $P_0 \longrightarrow^* P'$ and P is a subprocess of P'

Subject Reduction

a process is *initial* if it does not contain free variables and runtime syntax

a process P is *reachable* if there is a well-typed initial process P_0 such that $P_0 \longrightarrow^* P'$ and P is a subprocess of P'

If P is a reachable process and $\Gamma \vdash P \triangleright \Delta$ and $P \longrightarrow^* P'$, then $\Gamma \vdash P' \triangleright \Delta'$ for some Δ' such that $\Delta \Rightarrow^* \Delta'$

Communication Safety

evaluation contexts

$$\mathscr{E} \quad ::= \quad [-] \parallel \mathscr{E} \mid P \parallel \mathscr{E}; P \parallel (\nu a)\mathscr{E} \parallel (\nu s)\mathscr{E} \parallel s[\texttt{id} : r]!^{*}\langle \texttt{id}' : r', l\langle \mathscr{E} \rangle\rangle$$
$$\parallel \texttt{if } \mathscr{E} \texttt{ then } P \texttt{ else } P \parallel \mathscr{E} \wedge e \parallel v \wedge \mathscr{E} \parallel \ldots$$

Communication Safety

evaluation contexts

$$\mathscr{E} \quad ::= \quad [-] \parallel \mathscr{E} \mid P \parallel \mathscr{E}; P \parallel (\nu a)\mathscr{E} \parallel (\nu s)\mathscr{E} \parallel s[\mathtt{id} : r]!^{*}\langle \mathtt{id}' : r', l\langle \mathscr{E} \rangle\rangle \parallel \mathtt{if}\ \mathscr{E}\ \mathtt{then}\ P\ \mathtt{else}\ P \parallel \mathscr{E} \wedge e \parallel v \wedge \mathscr{E} \parallel \ldots$$

A process P is communication safe if:

- $P \equiv \mathscr{E}[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^{*} \mathscr{E}'[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v \rangle) \cdot \mathscr{B}]$ with $k \in I$;
- $P \equiv \mathscr{E}[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v \rangle) \cdot \mathscr{B}]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^{*} \mathscr{E}'[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ with $k \in I$

Communication Safety

A process P is *communication safe* if:

- $P \equiv \mathscr{E}[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^* \mathscr{E}'[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ with $k \in I$;
- $P \equiv \mathscr{E}[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^* \mathscr{E}'[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ with $k \in I$

join to restricted service names can destroy communication safety

Communication Safety

A process P is *communication safe* if:

- $P \equiv \mathscr{E}[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^* \mathscr{E}'[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ with $k \in I$;
- $P \equiv \mathscr{E}[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^* \mathscr{E}'[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ with $k \in I$

join to restricted service names can destroy communication safety

interleaving can destroy communication safety

Communication Safety

A process P is *communication safe* if:

- $P \equiv \mathscr{E}[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^* \mathscr{E}'[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ with $k \in I$;
- $P \equiv \mathscr{E}[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ implies that $\mathscr{E}[\mathbf{0}] \longrightarrow^* \mathscr{E}'[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ with $k \in I$

join to restricted service names can destroy communication safety

interleaving can destroy communication safety

we write $\Gamma \vdash^\star P \triangleright \Delta$ if this judgment can be obtained with a derivation where all session environments which appear in premises or conclusions of the system $\vdash_{\mathtt{id}}$ contain at most an association between a local type and a channel and joins can only be typed in the system $\vdash$

Communication Safety

A process P is *communication safe* if:

- $P \equiv \mathcal{E}[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ implies that $\mathcal{E}[\mathbf{0}] \longrightarrow^* \mathcal{E}'[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ with $k \in I$;
- $P \equiv \mathcal{E}[s : (\mathtt{id}' : r', \mathtt{id} : r, l_k\langle v\rangle) \cdot \mathscr{B}]$ implies that $\mathcal{E}[\mathbf{0}] \longrightarrow^* \mathcal{E}'[s[\mathtt{id} : r]?\langle \mathtt{id}' : r', \{l_i(\alpha_i).P_i\}_{i \in I}\rangle]$ with $k \in I$

we write $\Gamma \vdash^\star P \triangleright \Delta$ if this judgment can be obtained with a derivation where all session environments which appear in premises or conclusions of the system $\vdash_{\mathtt{id}}$ contain at most an association between a local type and a channel and joins can only be typed in the system $\vdash$

Let P be an initial process not containing restrictions. If $a : \langle G\rangle \vdash^\star P \triangleright \emptyset$ and $P \longrightarrow^* P'$, then P' is communication safe.

Progress

service registries are permanent

Progress

service registries are permanent and can always be reduced to produce sessions

Progress

service registries are permanent and can always be reduced to produce sessions

service joins can require conditions which are not satisfied

Progress

service registries are permanent and can always be reduced to produce sessions

service joins can require conditions which are not satisfied

A process P has the progress property if either P is a parallel composition of service registries and service joins, or there is P' such that $P \longrightarrow P'$ with a rule different from session initiation.

Progress

A process P has the *progress property* if either P is a parallel composition of service registries and service joins, or there is P' such that $P \longrightarrow P'$ with a rule different from session initiation.

Let $P \equiv a\langle G, \phi\rangle \mid P_0$ be an initial process not containing restrictions. If $a : \langle G\rangle \vdash^{\star} P \triangleright \emptyset$, then P has the progress property.

Local Types for Reputations

Restricting to reductions which activate at most one session for each service we can take advantage from local types for predicting possible future reputations of principals

Local Types for Reputations

Restricting to reductions which activate at most one session for each service we can take advantage from local types for predicting possible future reputations of principals

if n_r is the number of principals playing role r in the session, then the number of occurrences of label l (non occurring under recursion) in the histories of a role with local type T is bounded by $\#(T, l)$:

$$\#(!^{\bullet}\langle p, \{l_i\langle U_i\rangle.T_i\}_{i\in I}\rangle, l) = \begin{cases} max\{\#(T_{i_0}, l)+1, \#(T_i, l) \mid i \in I \setminus \{i_0\}\} & \text{if } l = l_{i_0} \ \&\ i_0 \in I, \\ max\{\#(T_i, l) \mid i \in I\} & \text{otherwise.} \end{cases}$$

$$\begin{array}{lcl@{\qquad}lcl}
\#(!\langle p, \{l_i\langle U_i\rangle.T_i\}_{i\in I}\rangle, l) & = & max\{\#(T_i, l) \mid i \in I\} & \#(\mu\mathbf{x}.T, l) & = & 0 \\
\#(?\langle p, \{l_i\langle U_i\rangle.T_i\}_{i\in I}\rangle, l) & = & max\{\#(T_i, l) \mid i \in I\} & \#(\mathbf{x}, l) & = & 0 \\
\#(\forall\ \iota : \mathtt{C}(r,\iota).T, l) & = & \#(T, l) \times n_r & \#(\epsilon, l) & = & 0 \\
\#(\exists\ \iota : \mathtt{C}(r,\iota).T, l) & = & \#(T, l) & \#(\mathrm{end}, l) & = & 0 \\
\#(T_1 \mid T_2, l) & = & \#(T_1, l) + \#(T_2, l) & \#(T_1\ ;\ T_2, l) & = & \#(T_1, l) + \#(T_2, l)
\end{array}$$

Local Types for Reputations

Restricting to reductions which activate at most one session for each service we can take advantage from local types for predicting possible future reputations of principals

we can bound the number of occurrences of label l in (part of) the histories to M by setting

$$\phi(ⓗ, r) \text{ equal to } m + \#(T, l) \leq M$$

where m is the number of occurrences of l in the considered part of ⓗ and type T is the local type of current role

Local Types for Reputations

Restricting to reductions which activate at most one session for each service we can take advantage from local types for predicting possible future reputations of principals

we can bound the number of occurrences of label l in (part of) the histories to M by setting

$$\phi(ⓗ, r) \text{ equal to } m + \#(T, l) \leq M$$

where m is the number of occurrences of l in the considered part of ⓗ and type T is the local type of current role
if we want to limit to 3 the number of rejections in the last 90 transitions, the joining condition for the *buyer* can hold true only if in the last 89 transitions the number of KO is less than or equal to 2.

Dynamic Multirole Session Types

Pierre-Malo Deniélou and Nobuko Yoshida. Dynamic Multirole Session Types. In *Proc. POPL'11*, pages 435–446. ACM, 2011.

Dynamic Multirole Session Types

Pierre-Malo Deniélou and Nobuko Yoshida. Dynamic Multirole Session Types. In *Proc. POPL'11*, pages 435–446. ACM, 2011.

Main differences:

Dynamic Multirole Session Types

Pierre-Malo Deniélou and Nobuko Yoshida. Dynamic Multirole Session Types. In *Proc. POPL'11*, pages 435–446. ACM, 2011.

Main differences:

1. we distinguish services and sessions, allowing multiple sessions for a single service;

Dynamic Multirole Session Types

Pierre-Malo Deniélou and Nobuko Yoshida. Dynamic Multirole Session Types. In *Proc. POPL'11*, pages 435–446. ACM, 2011.

Main differences:

1. we distinguish services and sessions, allowing multiple sessions for a single service;
2. we associate histories with principals;

Dynamic Multirole Session Types

Pierre-Malo Deniélou and Nobuko Yoshida. Dynamic Multirole Session Types. In *Proc. POPL'11*, pages 435–446. ACM, 2011.

Main differences:

1. we distinguish services and sessions, allowing multiple sessions for a single service;
2. we associate histories with principals;
3. we associate policies with joins, services and polls;

Dynamic Multirole Session Types

Pierre-Malo Deniélou and Nobuko Yoshida. Dynamic Multirole Session Types. In *Proc. POPL'11*, pages 435–446. ACM, 2011.

Main differences:

1. we distinguish services and sessions, allowing multiple sessions for a single service;
2. we associate histories with principals;
3. we associate policies with joins, services and polls;
4. we offer a primitive for choosing one of the best principals (according to some specified policy) among those playing a given role, if any.

Calculus for Trust Management

Marco Carbone, Mogens Nielsen, and Vladimiro Sassone. A Calculus of Trust Management. In *FSTTCS'04*, volume 3328 of *LNCS*, pages 161–173. Springer, 2004.

Calculus for Trust Management

Marco Carbone, Mogens Nielsen, and Vladimiro Sassone. A Calculus of Trust Management. In *FSTTCS'04*, volume 3328 of *LNCS*, pages 161–173. Springer, 2004.

Each principal has:

Calculus for Trust Management

Marco Carbone, Mogens Nielsen, and Vladimiro Sassone. A Calculus of Trust Management. In *FSTTCS'04*, volume 3328 of *LNCS*, pages 161–173. Springer, 2004.

Each principal has:

- a protocol (π-calculus process)

Calculus for Trust Management

Marco Carbone, Mogens Nielsen, and Vladimiro Sassone. A Calculus of Trust Management. In *FSTTCS'04*, volume 3328 of *LNCS*, pages 161–173. Springer, 2004.

Each principal has:

- a protocol (π-calculus process)
- a policy consisting of:

Calculus for Trust Management

Marco Carbone, Mogens Nielsen, and Vladimiro Sassone. A Calculus of Trust Management. In *FSTTCS'04*, volume 3328 of *LNCS*, pages 161–173. Springer, 2004.

Each principal has:

- a protocol (π-calculus process)
- a policy consisting of:
  - decision rules (logical formulas)

Calculus for Trust Management

Marco Carbone, Mogens Nielsen, and Vladimiro Sassone. A Calculus of Trust Management. In *FSTTCS'04*, volume 3328 of *LNCS*, pages 161–173. Springer, 2004.

Each principal has:

- a protocol (π-calculus process)
- a policy consisting of:
  - decision rules (logical formulas)
  - experiences (messages received from other participants)

Calculus for Trust Management

Marco Carbone, Mogens Nielsen, and Vladimiro Sassone. A Calculus of Trust Management. In *FSTTCS'04*, volume 3328 of *LNCS*, pages 161–173. Springer, 2004.

Each principal has:

- a protocol (π-calculus process)
- a policy consisting of:
  - decision rules (logical formulas)
  - experiences (messages received from other participants)

The focus is on barbed equivalence among principals

Conclusions

We presented a role-based multiparty session calculus that takes into account the history of principals, in order to measure their reputation and regulate accordingly their participation in future conversations.

Conclusions

We presented a role-based multiparty session calculus that takes into account the history of principals, in order to measure their reputation and regulate accordingly their participation in future conversations.

Future work:

Conclusions

We presented a role-based multiparty session calculus that takes into account the history of principals, in order to measure their reputation and regulate accordingly their participation in future conversations.

Future work:

- use reputation in order to deal with security;

Conclusions

We presented a role-based multiparty session calculus that takes into account the history of principals, in order to measure their reputation and regulate accordingly their participation in future conversations.

Future work:

- use reputation in order to deal with security;
- refine the notions of reputation and policies:

Conclusions

We presented a role-based multiparty session calculus that takes into account the history of principals, in order to measure their reputation and regulate accordingly their participation in future conversations.

Future work:

- use reputation in order to deal with security;
- refine the notions of reputation and policies:
  - extend histories;

Conclusions

We presented a role-based multiparty session calculus that takes into account the history of principals, in order to measure their reputation and regulate accordingly their participation in future conversations.

Future work:

- use reputation in order to deal with security;
- refine the notions of reputation and policies:
  - extend histories;
  - subjective reputations.

Conclusions

We presented a role-based multiparty session calculus that takes into account the history of principals, in order to measure their reputation and regulate accordingly their participation in future conversations.

Future work:

- use reputation in order to deal with security;
- refine the notions of reputation and policies:
  - extend histories;
  - subjective reputations.

Thank you!