Self-Adaptive Monitors for Multiparty Sessions

joint work with Mario Coppo and Betti Venneri

London, 10 January 2014

Adaptation

What is adaptation?

Adaptation

What is adaptation?

R. Bruni, A. Corradini, F. Gadducci, A. Lluch-Lafuente, and A. Vandin, "A Conceptual Framework for Adaptation," in *FASE'12*, ser. LNCS, vol. 7212. Springer, 2012, pp. 240–254.

Adaptation

What is adaptation?

R. Bruni, A. Corradini, F. Gadducci, A. Lluch-Lafuente, and A. Vandin, "A Conceptual Framework for Adaptation," in *FASE'12*, ser. LNCS, vol. 7212. Springer, 2012, pp. 240–254.

"we define adaptation as the run-time modification of the control data ...and a component is self-adaptive if it is able to modify its own control data at run-time"

Adaptation

What is adaptation?

R. Bruni, A. Corradini, F. Gadducci, A. Lluch-Lafuente, and A. Vandin, "A Conceptual Framework for Adaptation," in *FASE'12*, ser. LNCS, vol. 7212. Springer, 2012, pp. 240–254.

"we define adaptation as the run-time modification of the control data ...and a component is self-adaptive if it is able to modify its own control data at run-time"

we need to distinguish between standard data and control data: a change in the system behaviour is part of the application logic if it is based on standard data, it is an adaptation if it is based on control data.

Scenarios

- a community has many distributed entities which interact with each other according to a given operational plan,

Scenarios

- a community has many distributed entities which interact with each other according to a given operational plan,
- the complex dynamic environment can present unforeseen events, which require the community to modify its plan dynamically,

Scenarios

- a community has many distributed entities which interact with each other according to a given operational plan,
- the complex dynamic environment can present unforeseen events, which require the community to modify its plan dynamically,
- those critical events are observed in any separate component of the system, which can be checked by the session participants, so that the whole system can react promptly by updating itself,

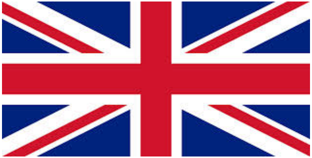
Scenarios

- a community has many distributed entities which interact with each other according to a given operational plan,
- the complex dynamic environment can present unforeseen events, which require the community to modify its plan dynamically,
- those critical events are observed in any separate component of the system, which can be checked by the session participants, so that the whole system can react promptly by updating itself,
- the dynamic changes need to be rather flexible: in each adaptation phase, new participants can be introduced or some of the old participants are not longer involved (temporarily or permanently),

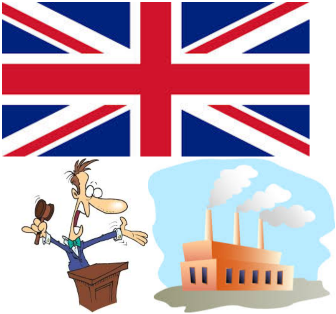
Scenarios

- a community has many distributed entities which interact with each other according to a given operational plan,
- the complex dynamic environment can present unforeseen events, which require the community to modify its plan dynamically,
- those critical events are observed in any separate component of the system, which can be checked by the session participants, so that the whole system can react promptly by updating itself,
- the dynamic changes need to be rather flexible: in each adaptation phase, new participants can be introduced or some of the old participants are not longer involved (temporarily or permanently),
- these dynamic changes need to be safe: community interactions must proceed correctly to pursue the common task.

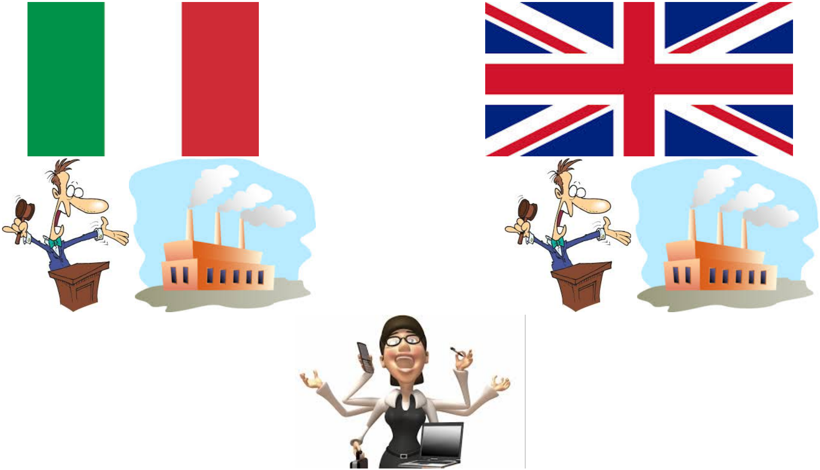
Example



Example



Example

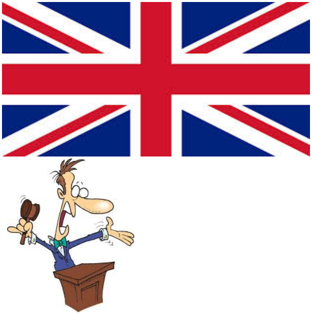


Global Type

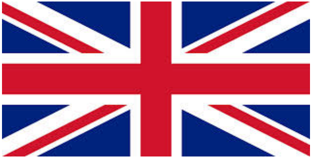
Global Type

$$G_1 = \begin{array}{r} \textsf{iS} \rightarrow \textsf{iF} : (\textsf{Item}, \textsf{Amount}). \\ \textsf{bS} \rightarrow \textsf{bF} : (\textsf{Item}, \textsf{Amount}). \\ \textsf{Ada} \rightarrow \{\textsf{iS}, \textsf{iF}, \textsf{bS}, \textsf{bF}\} : \textit{check} \end{array}$$

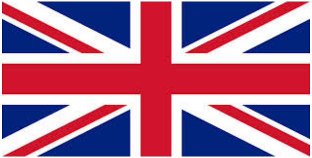
Example



Example



Example



Global Type

Global Type

$$G_2 = \begin{array}{c} \mathsf{Ada} \rightarrow \mathsf{Bob} : \mathsf{Contract}. \\ \mathsf{iS} \rightarrow \mathsf{iF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{bS} \rightarrow \mathsf{Ada} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{Ada} \rightarrow \{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{Bob}\} : \mathit{check} \end{array}$$

Example

$$\mathsf{iS} \rightarrow \mathsf{iF} : (\mathsf{Item}, \mathsf{Amount}).$$
$$\mathsf{bS} \rightarrow \mathsf{bF} : (\mathsf{Item}, \mathsf{Amount}).$$
$$\mathsf{Ada} \rightarrow \{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{bF}\} : \textit{check}$$

Monitors

$$G_1 = \begin{array}{rl} \mathsf{iS} \rightarrow & \mathsf{iF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{bS} \rightarrow & \mathsf{bF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{Ada} \rightarrow & \{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{bF}\} : \mathit{check} \end{array}$$

Monitors

$$G_1 = \begin{array}{r} \mathsf{iS} \to \mathsf{iF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{bS} \to \mathsf{bF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{Ada} \to \{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{bF}\} : \textit{check} \end{array}$$

 $\mathsf{iF}!(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\textit{check}$

 $\mathsf{bF}!(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\textit{check}$

Monitors

$$G_1 = \begin{array}{r}\textcolor{blue}{\mathsf{iS}} \rightarrow \textcolor{blue}{\mathsf{iF}} : (\mathsf{Item}, \mathsf{Amount}).\\ \textcolor{brown}{\mathsf{bS}} \rightarrow \textcolor{brown}{\mathsf{bF}} : (\mathsf{Item}, \mathsf{Amount}).\\ \mathsf{Ada} \rightarrow \{\textcolor{blue}{\mathsf{iS}}, \textcolor{blue}{\mathsf{iF}}, \textcolor{brown}{\mathsf{bS}}, \textcolor{brown}{\mathsf{bF}}\} : \textcolor{red}{\mathit{check}}\end{array}$$

 iF!(Item, Amount).Ada?*check*

 bF!(Item, Amount).Ada?*check*

 iS?(Item, Amount).Ada?*check*

 bS?(Item, Amount).Ada?*check*

Monitors

$$G_1 = \begin{array}{l} \mathsf{iS} \to \mathsf{iF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{bS} \to \mathsf{bF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{Ada} \to \{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{bF}\} : \mathit{check} \end{array}$$

 $\mathsf{iF}!(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\mathit{check}$

 $\mathsf{bF}!(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\mathit{check}$

 $\mathsf{iS}?(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\mathit{check}$

 $\mathsf{bS}?(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\mathit{check}$

 $\{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{bF}\}!\mathit{check}$

Monitors

$$G_2 = \begin{array}{c} \mathsf{Ada} \to \mathsf{Bob} : \mathsf{Contract}. \\ \mathsf{iS} \to \mathsf{iF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{bS} \to \mathsf{Ada} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{Ada} \to \{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{Bob}\} : \mathit{check} \end{array}$$

Monitors

$$G_2 = \begin{array}{c} \text{Ada} \to \text{Bob} : \text{Contract}. \\ \text{iS} \to \text{iF} : (\text{Item}, \text{Amount}). \\ \text{bS} \to \text{Ada} : (\text{Item}, \text{Amount}). \\ \text{Ada} \to \{\text{iS}, \text{iF}, \text{bS}, \text{Bob}\} : \textit{check} \end{array}$$

iF!(Item, Amount).Ada?*check*

Ada!(Item, Amount).Ada?*check*

Monitors

$$G_2 = \begin{array}{c} \text{Ada} \to \text{Bob} : \text{Contract}. \\ \color{blue}{\text{iS}} \to \color{blue}{\text{iF}} : (\text{Item}, \text{Amount}). \\ \color{brown}{\text{bS}} \to \text{Ada} : (\text{Item}, \text{Amount}). \\ \text{Ada} \to \{\color{blue}{\text{iS}}, \color{blue}{\text{iF}}, \color{brown}{\text{bS}}, \text{Bob}\} : \color{red}{\textit{check}} \end{array}$$


iF!(Item, Amount).Ada?*check*


Ada!(Item, Amount).Ada?*check*


iS?(Item, Amount).Ada?*check*

Monitors

$$G_2 = \begin{array}{c} \text{Ada} \to \text{Bob} : \text{Contract}. \\ \text{iS} \to \text{iF} : (\text{Item}, \text{Amount}). \\ \text{bS} \to \text{Ada} : (\text{Item}, \text{Amount}). \\ \text{Ada} \to \{\text{iS}, \text{iF}, \text{bS}, \text{Bob}\} : \textit{check} \end{array}$$

 iF!(Item, Amount).Ada?*check*

 Ada!(Item, Amount).Ada?*check*

 iS?(Item, Amount).Ada?*check*

 Bob!Contract.bS?(Item, Amount).{iS, iF, bS, Bob}!*check*

Monitors

$$G_2 = \begin{array}{rl} \mathsf{Ada} \to & \mathsf{Bob} : \mathsf{Contract}. \\ \mathsf{iS} \to & \mathsf{iF} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{bS} \to & \mathsf{Ada} : (\mathsf{Item}, \mathsf{Amount}). \\ \mathsf{Ada} \to & \{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{Bob}\} : \mathit{check} \end{array}$$

 $\mathsf{iF}!(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\mathit{check}$

 $\mathsf{Ada}!(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\mathit{check}$

 $\mathsf{iS}?(\mathsf{Item}, \mathsf{Amount}).\mathsf{Ada}?\mathit{check}$

 $\mathsf{Bob}!\mathsf{Contract}.\mathsf{bS}?(\mathsf{Item}, \mathsf{Amount}).\{\mathsf{iS}, \mathsf{iF}, \mathsf{bS}, \mathsf{Bob}\}!\mathit{check}$

 $\mathsf{Ada}?\mathsf{Contract}.\mathsf{Ada}?\mathit{check}$

Processes

$$\mu X.y!(\mathsf{item}, \mathsf{amount}).y?\mathit{check}.X$$

Processes

$\mu X.y!(\text{item}, \text{amount}).y?\mathit{check}.X$

$\mu X.y?(\text{item}, \text{amount}).\text{if} \ldots \text{ then } y?\mathit{check}.X$
$\text{else write KO}.y?\mathit{check}$

Processes

$$\mu X.y!(\text{item},\text{amount}).y?\mathit{check}.X$$

$$\mu X.y?(\text{item},\text{amount}).\text{if }\ldots\text{ then } y?\mathit{check}.X \quad \text{else write } \text{KO}.y?\mathit{check}$$

$$(\mu X.y!\mathit{check}(F).X) + (\mu X.y!\text{contract}.y?(\text{item},\text{amount}).y!\mathit{check}(F).X)$$

Processes

$$\mu X.y!(\mathrm{item}, \mathrm{amount}).y?\mathit{check}.X$$

$$\mu X.y?(\mathrm{item}, \mathrm{amount}).\mathrm{if}\ \ldots\ \mathrm{then}\ y?\mathit{check}.X$$
$$\mathrm{else\ write\ KO}.y?\mathit{check}$$

$$(\mu X.y!\mathit{check}(F).X)$$
$$+(\mu X.y!\mathrm{contract}.y?(\mathrm{item}, \mathrm{amount}).y!\mathit{check}(F).X)$$

$$\mu X.y?\mathrm{contract}.\mathrm{if}\ \ldots\ \mathrm{then\ write\ OK}.y?\mathit{check}$$
$$\mathrm{else}\ y?\mathit{check}.X$$

Processes

$\mu X.y!(\text{item}, \text{amount}).y?\textit{check}.X$

$\mu X.y?(\text{item}, \text{amount}).\text{if } \dots \text{ then } y?\textit{check}.X$
$\qquad \text{else write KO}.y?\textit{check}$

$(\mu X.y!\textit{check}(F).X)$
$+(\mu X.y!\text{contract}.y?(\text{item}, \text{amount}).y!\textit{check}(F).X)$

$\mu X.y?\text{contract}.\text{if } \dots \text{ then write OK}.y?\textit{check}$
$\qquad \text{else } y?\textit{check}.X$

$$F(\text{OK}, \text{OK}) = G_1 \quad F(\text{OK}, \text{KO}) = G_2$$
$$F(\text{KO}, \text{OK}) = G_3 \quad F(\text{KO}, \text{KO}) = G_4$$

System

$\text{iF}!(\text{Item}, \text{Amount}).\text{Ada}?\mathit{check} \quad [\mu X.y!(\text{item}, \text{amount}).y?\mathit{check}.X] \mid$

System

 $\textcolor{blue}{\mathrm{iF}}!(\mathrm{Item},\mathrm{Amount}).\mathrm{Ada}?\textcolor{red}{\mathit{check}}\quad[\mu X.y!(\mathrm{item},\mathrm{amount}).y?\textcolor{red}{\mathit{check}}.X]\ |$

 $\textcolor{brown}{\mathrm{bF}}!(\mathrm{Item},\mathrm{Amount}).\mathrm{Ada}?\textcolor{red}{\mathit{check}}\quad[\mu X.y!(\mathrm{item},\mathrm{amount}).y?\textcolor{red}{\mathit{check}}.X]\ |$

System

 $\color{blue}{\text{iF}}!(\text{Item}, \text{Amount}).\text{Ada}?\color{red}{\textit{check}}$ $[\mu X.y!(\text{item}, \text{amount}).y?\color{red}{\textit{check}}.X] \mid$

 $\text{bF}!(\text{Item}, \text{Amount}).\text{Ada}?\color{red}{\textit{check}}$ $[\mu X.y!(\text{item}, \text{amount}).y?\color{red}{\textit{check}}.X] \mid$

 $\color{blue}{\text{iS}}?(\text{Item}, \text{Amount}).\text{Ada}?\color{red}{\textit{check}}$ $[\mu X.y?(\text{item}, \text{amount}).\text{if } \ldots$

$\text{then } y?\color{red}{\textit{check}}.X$

$\text{else write } \color{red}{\text{KO}}.y?\color{red}{\textit{check}}] \mid$

System

	iF!(Item, Amount).Ada?*check*	$[\mu X.y!(\text{item}, \text{amount}).y?\mathit{check}.X] \mid$
	bF!(Item, Amount).Ada?*check*	$[\mu X.y!(\text{item}, \text{amount}).y?\mathit{check}.X] \mid$
	iS?(Item, Amount).Ada?*check*	$[\mu X.y?(\text{item}, \text{amount}).\text{if} \ldots$ then $y?\mathit{check}.X$ else write KO.$y?\mathit{check}] \mid$
	bS?(Item, Amount).Ada?*check*	$[\mu X.y?(\text{item}, \text{amount}).\text{if} \ldots$ then $y?\mathit{check}.X$ else write KO.$y?\mathit{check}] \mid$

System

	$\mathrm{iF}!(\mathrm{Item},\mathrm{Amount}).\mathrm{Ada}?\mathit{check}$	$[\mu X.y!(\mathrm{item},\mathrm{amount}).y?\mathit{check}.X] \mid$
	$\mathrm{bF}!(\mathrm{Item},\mathrm{Amount}).\mathrm{Ada}?\mathit{check}$	$[\mu X.y!(\mathrm{item},\mathrm{amount}).y?\mathit{check}.X] \mid$
	$\mathrm{iS}?(\mathrm{Item},\mathrm{Amount}).\mathrm{Ada}?\mathit{check}$	$[\mu X.y?(\mathrm{item},\mathrm{amount}).\mathrm{if}\ \ldots$ then $y?\mathit{check}.X$ else write $\mathrm{KO}.y?\mathit{check}] \mid$
	$\mathrm{bS}?(\mathrm{Item},\mathrm{Amount}).\mathrm{Ada}?\mathit{check}$	$[\mu X.y?(\mathrm{item},\mathrm{amount}).\mathrm{if}\ \ldots$ then $y?\mathit{check}.X$ else write $\mathrm{KO}.y?\mathit{check}] \mid$
	$\{\mathrm{iS},\mathrm{iF},\mathrm{bS},\mathrm{bF}\}!\mathit{check}$	$[(\mu X.y!\mathit{check}(F).X) + (\mu X.y!\mathrm{contract}.y?(\mathrm{item},\mathrm{amount}).y!\mathit{check}(F).X)] \parallel$

System

	iF!(Item, Amount).Ada?*check*	$[\mu X.y!(\text{item}, \text{amount}).y?\mathit{check}.X] \mid$
	bF!(Item, Amount).Ada?*check*	$[\mu X.y!(\text{item}, \text{amount}).y?\mathit{check}.X] \mid$
	iS?(Item, Amount).Ada?*check*	$[\mu X.y?(\text{item}, \text{amount}).\text{if } \ldots$ then $y?\mathit{check}.X$ else write KO.$y?\mathit{check}] \mid$
	bS?(Item, Amount).Ada?*check*	$[\mu X.y?(\text{item}, \text{amount}).\text{if } \ldots$ then $y?\mathit{check}.X$ else write KO.$y?\mathit{check}] \mid$
	{iS, iF, bS, bF}!*check*	$[(\mu X.y!\mathit{check}(F).X) + (\mu X.y!\text{contract}.y?(\text{item}, \text{amount}).y!\mathit{check}(F).X)] \parallel$

(OK, OK)

Syntax

Global types

$$G \quad ::= \quad \mathsf{p} \to \Pi : \{\ell_i(S_i).G_i\}_{i\in I} \quad \| \quad \mathsf{p} \to \Pi : \{\lambda_i\}_{i\in I} \quad \| \quad \mathsf{end}$$

Syntax

Global types

$$G ::= \mathsf{p} \to \Pi : \{\ell_i(S_i).G_i\}_{i\in I} \quad \| \quad \mathsf{p} \to \Pi : \{\lambda_i\}_{i\in I} \quad \| \quad \mathsf{end}$$

Monitors

$$\mathcal{M} ::= \mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \quad \| \quad \Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \quad \| \quad \mathsf{p}?\{\lambda_i\}_{i\in I} \quad \| \quad \Pi!\{\lambda_i\}_{i\in I} \quad \| \quad \mathsf{end}$$

Syntax

Global types

$$G \quad ::= \quad \mathsf{p} \to \Pi : \{\ell_i(S_i).G_i\}_{i \in I} \quad | \quad \mathsf{p} \to \Pi : \{\lambda_i\}_{i \in I} \quad | \quad \mathsf{end}$$

Monitors

$$\mathcal{M} \quad ::= \quad \mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i \in I} \quad | \quad \Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i \in I} \quad | \quad \mathsf{p}?\{\lambda_i\}_{i \in I} \quad | \quad \Pi!\{\lambda_i\}_{i \in I} \quad | \quad \mathsf{end}$$

Processes

$$P \quad ::= \quad \mathbf{0} \quad | \quad \mathsf{op}.P \quad | \quad X \quad | \quad \mu X.P \quad | \quad \mathsf{c}?\ell(x).P \quad | \quad \mathsf{c}!\ell(\mathsf{e}).P \quad | \quad \mathsf{c}?(\lambda, \mathsf{T}).P \quad | \quad \mathsf{c}!(\lambda(F), \mathsf{T}).P \quad | \quad \mathsf{if}\ \mathsf{e}\ \mathsf{then}\ P\ \mathsf{else}\ P \quad | \quad P + P$$

Syntax

Global types

$$G \quad ::= \quad \mathsf{p} \to \Pi : \{\ell_i(S_i).G_i\}_{i\in I} \quad \| \quad \mathsf{p} \to \Pi : \{\lambda_i\}_{i\in I} \quad \| \quad \mathsf{end}$$

Monitors

$$\mathcal{M} \quad ::= \quad \mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \quad \| \quad \Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \quad \| \quad \mathsf{p}?\{\lambda_i\}_{i\in I} \quad \| \quad \Pi!\{\lambda_i\}_{i\in I} \quad \| \quad \mathsf{end}$$

Processes

$$P \quad ::= \quad \mathbf{0} \quad \| \quad \mathsf{op}.P \quad \| \quad X \quad \| \quad \mu X.P \quad \| \quad \mathsf{c}?\ell(x).P \quad \| \quad \mathsf{c}!\ell(\mathsf{e}).P \quad \| \quad \mathsf{c}?(\lambda, \mathsf{T}).P \quad \| \quad \mathsf{c}!(\lambda(F), \mathsf{T}).P \quad \| \quad \mathsf{if}\ \mathsf{e}\ \mathsf{then}\ P\ \mathsf{else}\ P \quad \| \quad P+P$$

Networks

$$N \quad ::= \quad \mathsf{new}(G) \quad \| \quad \mathcal{M}[P] \quad \| \quad \mathsf{s}:h \quad \| \quad N \mid N \quad \| \quad (\nu \mathsf{s})N$$

Syntax

Global types

$$G \quad ::= \quad \mathsf{p} \to \Pi : \{\ell_i(S_i).G_i\}_{i\in I} \quad \| \quad \mathsf{p} \to \Pi : \{\lambda_i\}_{i\in I} \quad \| \quad \mathsf{end}$$

Monitors

$$\mathcal{M} \quad ::= \quad \mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \quad \| \quad \Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \quad \| \quad \mathsf{p}?\{\lambda_i\}_{i\in I} \quad \| \quad \Pi!\{\lambda_i\}_{i\in I} \quad \| \quad \mathsf{end}$$

Processes

$$P \quad ::= \quad \mathbf{0} \quad \| \quad \mathsf{op}.P \quad \| \quad X \quad \| \quad \mu X.P \quad \| \quad \mathsf{c}?\ell(x).P \quad \| \quad \mathsf{c}!\ell(\mathsf{e}).P \quad \| \quad \mathsf{c}?(\lambda, \mathsf{T}).P \quad \| \quad \mathsf{c}!(\lambda(F), \mathsf{T}).P \quad \| \quad \mathsf{if}\ \mathsf{e}\ \mathsf{then}\ P\ \mathsf{else}\ P \quad \| \quad P+P$$

Networks

$$N \quad ::= \quad \mathsf{new}(G) \quad \| \quad \mathcal{M}[P] \quad \| \quad \mathsf{s}:h \quad \| \quad N \mid N \quad \| \quad (\nu \mathsf{s})N$$

Systems

$$\mathcal{S} \quad ::= \quad N \parallel \sigma$$

Process Types

$$T \;::=\; ?\ell(S).T \mid !\ell(S).T \mid ?\lambda \mid !\lambda \mid T \wedge T \mid T \vee T \mid \mathsf{end}$$

Process Types

$$T \;::=\; ?\ell(S).T \parallel !\ell(S).T \parallel ?\lambda \parallel !\lambda \parallel T \wedge T \parallel T \vee T \parallel \mathsf{end}$$

$$\Gamma, X : \mathsf{T} \vdash \mathsf{c}?(\lambda, \mathsf{T}).X \triangleright \mathsf{c} : ?\lambda \qquad \Gamma, X : \mathsf{T} \vdash \mathsf{c}!(\lambda(F), \mathsf{T}).X \triangleright \mathsf{c} : !\lambda$$

Process Types

$$T \ ::= \ ?\ell(S).T \mid !\ell(S).T \mid ?\lambda \mid !\lambda \mid T \wedge T \mid T \vee T \mid \text{end}$$

$$\Gamma, X : \mathsf{T} \vdash \mathsf{c}?(\lambda, \mathsf{T}).X \triangleright \mathsf{c} : ?\lambda \qquad \Gamma, X : \mathsf{T} \vdash \mathsf{c}!(\lambda(F), \mathsf{T}).X \triangleright \mathsf{c} : !\lambda$$

$$\frac{\Gamma \vdash P \triangleright \mathsf{c} : \mathsf{T}}{\Gamma \vdash \mathsf{c}?(\lambda, \mathsf{T}).P \triangleright \mathsf{c} : ?\lambda} \qquad \frac{\Gamma \vdash P \triangleright \mathsf{c} : \mathsf{T}}{\Gamma \vdash \mathsf{c}!(\lambda(F), \mathsf{T}).P \triangleright \mathsf{c} : !\lambda}$$

Process Types

$$T \ ::= \ ?\ell(S).T \mid !\ell(S).T \mid ?\lambda \mid !\lambda \mid T \wedge T \mid T \vee T \mid \mathsf{end}$$

$$\Gamma, X : \mathsf{T} \vdash \mathsf{c}?(\lambda, \mathsf{T}).X \triangleright \mathsf{c}:?\lambda \qquad \Gamma, X : \mathsf{T} \vdash \mathsf{c}!(\lambda(F), \mathsf{T}).X \triangleright \mathsf{c}:!\lambda$$

$$\frac{\Gamma \vdash P \triangleright \mathsf{c}:\mathsf{T}}{\Gamma \vdash \mathsf{c}?(\lambda, \mathsf{T}).P \triangleright \mathsf{c}:?\lambda} \qquad \frac{\Gamma \vdash P \triangleright \mathsf{c}:\mathsf{T}}{\Gamma \vdash \mathsf{c}!(\lambda(F), \mathsf{T}).P \triangleright \mathsf{c}:!\lambda}$$

$$\frac{\Gamma \vdash \mathsf{e} : \mathsf{bool} \quad \Gamma \vdash P_1 \triangleright \mathsf{c}:\mathsf{T}_1 \quad \Gamma \vdash P_2 \triangleright \mathsf{c}:\mathsf{T}_2 \quad \mathsf{T}_1 \vee \mathsf{T}_2 \in \mathcal{T}}{\Gamma \vdash \mathsf{if\ e\ then}\ P_1\ \mathsf{else}\ P_2 \triangleright \mathsf{c}:\mathsf{T}_1 \vee \mathsf{T}_2}$$

Process Types

$$T \;::=\; ?\ell(S).T \mid !\ell(S).T \mid ?\lambda \mid !\lambda \mid T \wedge T \mid T \vee T \mid \mathsf{end}$$

$$\Gamma, X : \mathsf{T} \vdash \mathsf{c}?(\lambda, \mathsf{T}).X \rhd \mathsf{c}:?\lambda \qquad \Gamma, X : \mathsf{T} \vdash \mathsf{c}!(\lambda(F), \mathsf{T}).X \rhd \mathsf{c}:!\lambda$$

$$\frac{\Gamma \vdash P \rhd \mathsf{c}:\mathsf{T}}{\Gamma \vdash \mathsf{c}?(\lambda, \mathsf{T}).P \rhd \mathsf{c}:?\lambda} \qquad \frac{\Gamma \vdash P \rhd \mathsf{c}:\mathsf{T}}{\Gamma \vdash \mathsf{c}!(\lambda(F), \mathsf{T}).P \rhd \mathsf{c}:!\lambda}$$

$$\frac{\Gamma \vdash \mathsf{e} : \mathsf{bool} \quad \Gamma \vdash P_1 \rhd \mathsf{c}:\mathsf{T}_1 \quad \Gamma \vdash P_2 \rhd \mathsf{c}:\mathsf{T}_2 \quad \mathsf{T}_1 \vee \mathsf{T}_2 \in \mathcal{T}}{\Gamma \vdash \mathsf{if\ e\ then}\ P_1\ \mathsf{else}\ P_2 \rhd \mathsf{c}:\mathsf{T}_1 \vee \mathsf{T}_2}$$

$$\frac{\Gamma \vdash P_1 \rhd \mathsf{c}:\mathsf{T}_1 \quad \Gamma \vdash P_2 \rhd \mathsf{c}:\mathsf{T}_2 \quad \mathsf{T}_1 \wedge \mathsf{T}_2 \in \mathcal{T}}{\Gamma \vdash P_1 + P_2 \rhd \mathsf{c}:\mathsf{T}_1 \wedge \mathsf{T}_2}$$

Adequacy of Types for Monitors

$$|\mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I}| = \bigwedge_{i\in I} ?\ell_i(S_i).|\mathcal{M}_i|$$

$$|\Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I}| = \bigvee_{i\in I} !\ell_i(S_i).|\mathcal{M}_i|$$

$$|\mathsf{p}?\{\lambda_i\}_{i\in I}| = \bigwedge_{i\in I} ?\lambda_i \qquad |\Pi!\{\lambda_i\}_{i\in I}| = \bigvee_{i\in I} !\lambda_i$$

$$|\mathsf{end}| = \mathsf{end}$$

Adequacy of Types for Monitors

$$|\mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I}| = \bigwedge_{i\in I} ?\ell_i(S_i).|\mathcal{M}_i|$$

$$|\Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I}| = \bigvee_{i\in I} !\ell_i(S_i).|\mathcal{M}_i|$$

$$|\mathsf{p}?\{\lambda_i\}_{i\in I}| = \bigwedge_{i\in I} ?\lambda_i \qquad |\Pi!\{\lambda_i\}_{i\in I}| = \bigvee_{i\in I} !\lambda_i$$

$$|\mathsf{end}| = \mathsf{end}$$

$$\mathsf{T} \leq \mathsf{end} \quad \mathsf{T}_1 \wedge \mathsf{T}_2 \ \leq \ \mathsf{T}_i \quad \mathsf{T}_i \leq \mathsf{T}_1 \vee \mathsf{T}_2 \quad (i = 1, 2)$$

$$\mathsf{T}_1 \leq \mathsf{T}_2 \ \text{ implies } \ !\ell(S).\mathsf{T}_1 \leq !\ell(S).\mathsf{T}_2 \quad ?\ell(S).\mathsf{T}_1 \leq ?\ell(S).\mathsf{T}_2$$

$$\mathsf{T} \leq \mathsf{T}_1 \text{ and } \mathsf{T} \leq \mathsf{T}_2 \text{ imply } \mathsf{T} \ \leq \ \mathsf{T}_1 \wedge \mathsf{T}_2$$

$$\mathsf{T}_1 \leq \mathsf{T} \text{ and } \mathsf{T}_2 \leq \mathsf{T} \text{ imply } \mathsf{T}_1 \vee \mathsf{T}_2 \leq \mathsf{T}$$

$$(\mathsf{T}_1 \vee \mathsf{T}_2) \wedge \mathsf{T}_3 = (\mathsf{T}_1 \wedge \mathsf{T}_3) \vee (\mathsf{T}_2 \wedge \mathsf{T}_3)$$

$$(\mathsf{T}_1 \wedge \mathsf{T}_2) \vee \mathsf{T}_3 = (\mathsf{T}_1 \vee \mathsf{T}_3) \wedge (\mathsf{T}_2 \vee \mathsf{T}_3)$$

Adequacy of Types for Monitors

$$|\mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I}| = \bigwedge_{i\in I} ?\ell_i(S_i).|\mathcal{M}_i|$$

$$|\Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I}| = \bigvee_{i\in I} !\ell_i(S_i).|\mathcal{M}_i|$$

$$|\mathsf{p}?\{\lambda_i\}_{i\in I}| = \bigwedge_{i\in I} ?\lambda_i \qquad |\Pi!\{\lambda_i\}_{i\in I}| = \bigvee_{i\in I} !\lambda_i$$

$$|\mathsf{end}| = \mathsf{end}$$

$$\mathsf{T} \leq \mathsf{end} \quad \mathsf{T}_1 \wedge \mathsf{T}_2 \ \leq \ \mathsf{T}_i \quad \mathsf{T}_i \leq \mathsf{T}_1 \vee \mathsf{T}_2 \quad (i = 1, 2)$$

$$\mathsf{T}_1 \leq \mathsf{T}_2 \ \text{ implies } \ !\ell(S).\mathsf{T}_1 \leq !\ell(S).\mathsf{T}_2 \quad ?\ell(S).\mathsf{T}_1 \leq ?\ell(S).\mathsf{T}_2$$

$$\mathsf{T} \leq \mathsf{T}_1 \text{ and } \mathsf{T} \leq \mathsf{T}_2 \text{ imply } \mathsf{T} \ \leq \ \mathsf{T}_1 \wedge \mathsf{T}_2$$

$$\mathsf{T}_1 \leq \mathsf{T} \text{ and } \mathsf{T}_2 \leq \mathsf{T} \text{ imply } \mathsf{T}_1 \vee \mathsf{T}_2 \leq \mathsf{T}$$

$$(\mathsf{T}_1 \vee \mathsf{T}_2) \wedge \mathsf{T}_3 = (\mathsf{T}_1 \wedge \mathsf{T}_3) \vee (\mathsf{T}_2 \wedge \mathsf{T}_3)$$

$$(\mathsf{T}_1 \wedge \mathsf{T}_2) \vee \mathsf{T}_3 = (\mathsf{T}_1 \vee \mathsf{T}_3) \wedge (\mathsf{T}_2 \vee \mathsf{T}_3)$$

A type T is adequate for a monitor $\mathcal{M}$ ($\mathsf{T} \propto \mathcal{M}$) if $\mathsf{T} \leq |\mathcal{M}|$.

Operational Semantics

LTS for monitors

$$\mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \xrightarrow{\mathsf{p}?\ell_j} \mathcal{M}_j \quad \Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \xrightarrow{\Pi!\ell_j} \mathcal{M}_j \quad j \in I$$

$$\mathsf{p}?\{\lambda_i\}_{i\in I} \xrightarrow{\mathsf{p}?\lambda_j} \quad \Pi!\{\lambda_i\}_{i\in I} \xrightarrow{\Pi!\lambda_j} \quad j \in I$$

Operational Semantics

LTS for monitors

$$\mathrm{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \xrightarrow{\mathrm{p}?\ell_j} \mathcal{M}_j \quad \Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \xrightarrow{\Pi!\ell_j} \mathcal{M}_j \quad j \in I$$

$$\mathrm{p}?\{\lambda_i\}_{i\in I} \xrightarrow{\mathrm{p}?\lambda_j} \quad \Pi!\{\lambda_i\}_{i\in I} \xrightarrow{\Pi!\lambda_j} \quad j \in I$$

LTS for processes

$$\mathrm{s[p]}?\ell(x).P \xrightarrow{\mathrm{s[p]}?\ell(v)} P\{v/x\} \quad \mathrm{s[p]}!\ell(\mathrm{e}).P \xrightarrow{\mathrm{s[p]}!\ell(v)} P \quad \mathrm{e}\downarrow v$$

$$\mathrm{s[p]}?(\lambda,\mathrm{T}).P \xrightarrow{\mathrm{s[p]}?(\lambda,\mathrm{T})} P \quad \mathrm{s[p]}!(\lambda(F),\mathrm{T}).P \xrightarrow{\mathrm{s[p]}!(\lambda(F),\mathrm{T})} P$$

Operational Semantics

LTS for monitors

$$\mathsf{p}?\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \xrightarrow{\mathsf{p}?\ell_j} \mathcal{M}_j \quad \Pi!\{\ell_i(S_i).\mathcal{M}_i\}_{i\in I} \xrightarrow{\Pi!\ell_j} \mathcal{M}_j \quad j \in I$$

$$\mathsf{p}?\{\lambda_i\}_{i\in I} \xrightarrow{\mathsf{p}?\lambda_j} \quad \Pi!\{\lambda_i\}_{i\in I} \xrightarrow{\Pi!\lambda_j} \quad j \in I$$

LTS for processes

$$\mathsf{s}[\mathsf{p}]?\ell(x).P \xrightarrow{\mathsf{s}[\mathsf{p}]?\ell(v)} P\{v/x\} \quad \mathsf{s}[\mathsf{p}]!\ell(\mathsf{e}).P \xrightarrow{\mathsf{s}[\mathsf{p}]!\ell(v)} P \quad \mathsf{e}\downarrow v$$

$$\mathsf{s}[\mathsf{p}]?(\lambda,\mathsf{T}).P \xrightarrow{\mathsf{s}[\mathsf{p}]?(\lambda,\mathsf{T})} P \quad \mathsf{s}[\mathsf{p}]!(\lambda(F),\mathsf{T}).P \xrightarrow{\mathsf{s}[\mathsf{p}]!(\lambda(F),\mathsf{T})} P$$

$$\frac{\Pi = \mathsf{pa}(G) \quad \mathcal{M}_{\mathsf{p}} = G\restriction\mathsf{p} \quad \forall \mathsf{p}\in\Pi.\ (P_{\mathsf{p}},\mathsf{T}_{\mathsf{p}}) \in \mathcal{P}\ \&\ \mathsf{T}_{\mathsf{p}} \propto \mathcal{M}_{\mathsf{p}}}{\mathsf{new}(G) \longrightarrow (\nu\,\mathsf{s})\ (\prod_{\mathsf{p}\in\Pi} \mathcal{M}_{\mathsf{p}}[P_{\mathsf{p}}\{\mathsf{s}[\mathsf{p}]/y\} \mid \mathsf{s} : \emptyset)}$$

Operational Semantics

$$\frac{\mathcal{M} \xrightarrow{\mathsf{q}?\ell} \mathcal{M}' \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]?\ell(v)} P'}{\mathcal{M}[P] \mid \mathsf{s} : (\mathsf{q}, \mathsf{p}, \ell(v)) \cdot h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h}$$

Operational Semantics

$$\frac{\mathcal{M} \xrightarrow{\mathsf{q}?\ell} \mathcal{M}' \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]?\ell(v)} P'}{\mathcal{M}[P] \mid \mathsf{s} : (\mathsf{q}, \mathsf{p}, \ell(v)) \cdot h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h}$$

$$\frac{\mathcal{M} \xrightarrow{\Pi!\ell} \mathcal{M}' \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]!\ell(v)} P'}{\mathcal{M}[P] \mid \mathsf{s} : h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h \cdot (\mathsf{p}, \Pi, \ell(v))}$$

Operational Semantics

$$\frac{\mathcal{M} \xrightarrow{\mathsf{q}?\ell} \mathcal{M}' \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]?\ell(v)} P'}{\mathcal{M}[P] \mid \mathsf{s} : (\mathsf{q}, \mathsf{p}, \ell(v)) \cdot h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h}$$

$$\frac{\mathcal{M} \xrightarrow{\Pi!\ell} \mathcal{M}' \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]!\ell(v)} P'}{\mathcal{M}[P] \mid \mathsf{s} : h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h \cdot (\mathsf{p}, \Pi, \ell(v))}$$

$$\frac{\mathcal{M} \xrightarrow{\mathsf{q}?\lambda} \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]?(\lambda,\mathsf{T})} P' \quad G \upharpoonright \mathsf{p} = \mathcal{M}' \qquad \mathsf{T} \propto \mathcal{M}'}{\mathcal{M}[P] \mid \mathsf{s} : (\mathsf{q}, \mathsf{p}, \lambda(G)) \cdot h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h}$$

Operational Semantics

$$\frac{\mathcal{M} \xrightarrow{\mathsf{q}?\ell} \mathcal{M}' \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]?\ell(v)} P'}{\mathcal{M}[P] \mid \mathsf{s} : (\mathsf{q}, \mathsf{p}, \ell(v)) \cdot h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h}$$

$$\frac{\mathcal{M} \xrightarrow{\Pi!\ell} \mathcal{M}' \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]!\ell(v)} P'}{\mathcal{M}[P] \mid \mathsf{s} : h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h \cdot (\mathsf{p}, \Pi, \ell(v))}$$

$$\frac{\mathcal{M} \xrightarrow{\mathsf{q}?\lambda} \qquad P \xrightarrow{\mathsf{s}[\mathsf{p}]?(\lambda,\mathsf{T})} P' \quad G \restriction \mathsf{p} = \mathcal{M}' \qquad \mathsf{T} \propto \mathcal{M}'}{\mathcal{M}[P] \mid \mathsf{s} : (\mathsf{q}, \mathsf{p}, \lambda(G)) \cdot h \longrightarrow \mathcal{M}'[P'] \mid \mathsf{s} : h}$$

$$\frac{\begin{array}{c}\mathcal{M} \xrightarrow{\Pi!\lambda} \quad P \xrightarrow{\mathsf{s}[\mathsf{p}]!(\lambda(F),\mathsf{T})} P' \quad F(\sigma) = G \quad \mathcal{M}_\mathsf{p} = G \restriction \mathsf{p} \quad \mathsf{T} \propto \mathcal{M}_\mathsf{p} \quad h\ \lambda\text{-free} \\ \Pi' = \mathsf{pa}(G) \quad \forall \mathsf{q} \in \Pi'.\mathcal{M}_\mathsf{q} = G \restriction \mathsf{q} \\ \forall \mathsf{q} \in \Pi' \setminus (\Pi \cup \{\mathsf{p}\}).\ (P_\mathsf{q}, \mathsf{T}_\mathsf{q}) \in \mathcal{P}\ \&\ \mathsf{T}_\mathsf{q} \propto \mathcal{M}_\mathsf{q}\end{array}}{\begin{array}{rl}\mathcal{M}[P] \mid \mathsf{s} : h \parallel \sigma \longrightarrow & \mathcal{M}_\mathsf{p}[P'] \mid \prod_{\mathsf{q} \in \Pi' \setminus (\Pi \cup \{\mathsf{p}\})} \mathcal{M}_\mathsf{q}[P_\mathsf{q}\{\mathsf{s}[\mathsf{q}]/y_\mathsf{q}\}] \mid \\ & \mathsf{s} : h \cdot (\mathsf{p}, \Pi, \lambda(G)) \parallel \sigma\end{array}}$$

Progress

a system whose network is a parallel composition of session initiators is initial

Progress

a system whose network is a parallel composition of session initiators is initial

a collection $\mathcal{P}$ is complete if, for every global type G in the domain of an adaptation function which occurs in a process belonging to $\mathcal{P}$, there are processes in $\mathcal{P}$ whose types are adequate for the monitors obtained by projecting G onto its participants

Progress

a system whose network is a parallel composition of session initiators is initial

a collection $\mathcal{P}$ is complete if, for every global type G in the domain of an adaptation function which occurs in a process belonging to $\mathcal{P}$, there are processes in $\mathcal{P}$ whose types are adequate for the monitors obtained by projecting G onto its participants

If $\mathcal{P}$ is complete, $\mathcal{S}$ is an initial system and $\mathcal{S} \longrightarrow^{*}_{\mathcal{P}} \mathcal{S}'$, then $\mathcal{S}'$ has progress, i.e.

1. every input monitored process will always (eventually) receive a message, and
2. every message in a queue will always (eventually) be received by an input monitored process.

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main ingredients:

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main ingredients:

- global types representing the overall communication choreography

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main ingredients:

- global types representing the overall communication choreography
- monitors adapting the behaviour of processes to the prescriptions of global types

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main ingredients:

- global types representing the overall communication choreography
- monitors adapting the behaviour of processes to the prescriptions of global types
- a global state implementing the control data

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main features:

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main features:

- the association of a monitor with a compliant process incarnates a single participant

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main features:

- the association of a monitor with a compliant process incarnates a single participant
- the choreography is updated at runtime, in response to changing conditions in the global state

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main features:

- the association of a monitor with a compliant process incarnates a single participant
- the choreography is updated at runtime, in response to changing conditions in the global state
- a decentralised control of the adaptation: any participant can be in charge of checking global data and sending the adaptation request

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main features:

- the association of a monitor with a compliant process incarnates a single participant
- the choreography is updated at runtime, in response to changing conditions in the global state
- a decentralised control of the adaptation: any participant can be in charge of checking global data and sending the adaptation request
- the dynamic system reconfiguration can add new participants, while some of the old participants are not longer involved

Summary

Focus: self-adaptiveness in the context of multiparty sessions

Main features:

- the association of a monitor with a compliant process incarnates a single participant
- the choreography is updated at runtime, in response to changing conditions in the global state
- a decentralised control of the adaptation: any participant can be in charge of checking global data and sending the adaptation request
- the dynamic system reconfiguration can add new participants, while some of the old participants are not longer involved
- processes, that are simply implementation code, can follow different incompatible computational paths

Related Papers

- T.-C. Chen, L. Bocchi, P.-M. Deniélou, K. Honda, and N. Yoshida, "Asynchronous Distributed Monitoring for Multiparty Session Enforcement": the monitors prescribe not only the types of the exchanged data, but also that the values of these data satisfy some predicates

Related Papers

- T.-C. Chen, L. Bocchi, P.-M. Deniélou, K. Honda, and N. Yoshida, "Asynchronous Distributed Monitoring for Multiparty Session Enforcement": the monitors prescribe not only the types of the exchanged data, but also that the values of these data satisfy some predicates
- G. Anderson and J. Rathke, "Dynamic Software Update for Message Passing Programs": global and session types are used to guarantee deadlock-freedom in a calculus of multiparty sessions with asynchronous communications

Related Papers

- T.-C. Chen, L. Bocchi, P.-M. Deniélou, K. Honda, and N. Yoshida, "Asynchronous Distributed Monitoring for Multiparty Session Enforcement": the monitors prescribe not only the types of the exchanged data, but also that the values of these data satisfy some predicates
- G. Anderson and J. Rathke, "Dynamic Software Update for Message Passing Programs": global and session types are used to guarantee deadlock-freedom in a calculus of multiparty sessions with asynchronous communications
- M. dalla Preda, I. Lanese, J. Mauro, M. Gabbrielli, and S. Giallorenzo, "Safe Run-time Adaptation of Distributed Systems": proposes a rule-based approach in which all interactions, under all possible changes produced by the adaptation rules, proceed as prescribed by an abstract model

Future Work

- experiment with implementations of our approach, to evaluate its feasibility

Future Work

- experiment with implementations of our approach, to evaluate its feasibility
- allow the global state to contain dynamically evolving semantic information about processes, such as reputation or performance rates

Future Work

- experiment with implementations of our approach, to evaluate its feasibility
- allow the global state to contain dynamically evolving semantic information about processes, such as reputation or performance rates
- exploit a consensus algorithm for choosing a single one among all the processes matching a monitor, using rates of processes

Future Work

- experiment with implementations of our approach, to evaluate its feasibility
- allow the global state to contain dynamically evolving semantic information about processes, such as reputation or performance rates
- exploit a consensus algorithm for choosing a single one among all the processes matching a monitor, using rates of processes