Towards Asynchronous Communication in Team Automata

Maurice ter Beek and José Proença

FMT, CNR–ISTI, Pisa, Italy

CISTER & U. Porto, Portugal

APM 2025, Porto, Portugal, October 3rd, 2024

Outline

- History of Team Automata
  - Motivation and development
  - Constrained multi-party synchronisations
  - Related coordination models
- Recent Results
  - **ICTAC'20** Extended Team Automata
  - **FM'21** Featured Team Automata
  - **FM'23** Model Check Team Automata
  - **ICTAC'23** Realisable Team Automata
- Ongoing Work: Asynchronous Team Automata
  - Asynchronous multi-party communication?
  - Safe communication in asynchronous setting?
  - Realisability of asynchronous team automata?
  - Tooling for asynchronous team automata?

History of Team Automata

Team Automata

A formalism for interacting component-based systems, whereby multiple sending and receiving actions from concurrent automata can synchronise on certain executions

First proposed at the 1997 ACM SIGGROUP Conference on Supporting Group Work for modelling components of groupware systems and their interconnections [GROUP'97]

Inspired by Input/Output (I/O) automata, inheriting the distinction between internal and external (input and output) actions used for communication with the environment

Team Automata

A formalism for interacting component-based systems, whereby multiple sending and receiving actions from concurrent automata can synchronise on certain executions

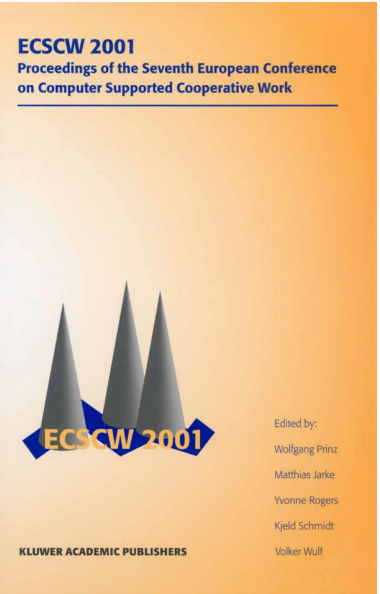
First proposed at the 1997 ACM SIGGROUP Conference on Supporting Group Work for modelling components of groupware systems and their interconnections [GROUP'97]

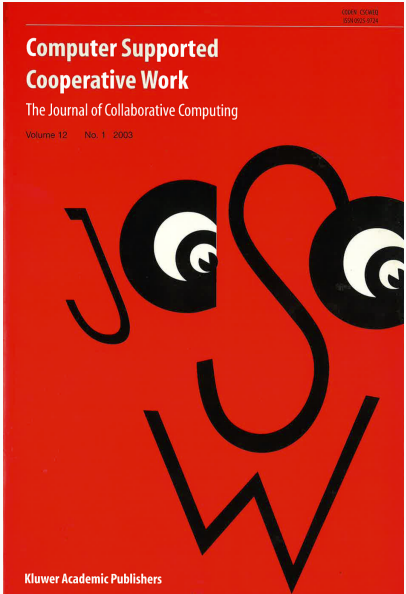
Inspired by Input/Output (I/O) automata, inheriting the distinction between internal and external (input and output) actions used for communication with the environment

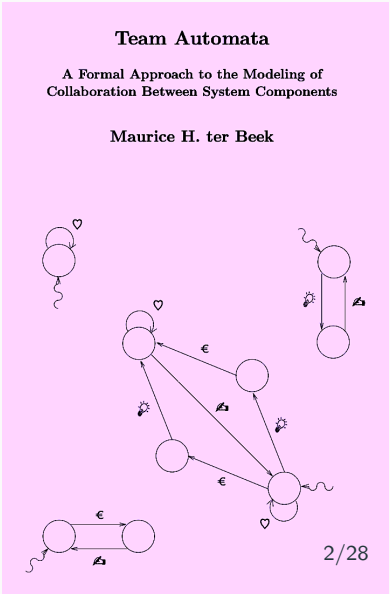
Formally defined in Computer Supported Cooperative Work (CSCW) — The Journal of Collaborative Computing, as composed by component automata that synchronise [CSCW'03]

Technically an extension of I/O automata, imposing hardly any restrictions on the role of actions in components, while composition is not limited to the synchronous product [IPL'05]

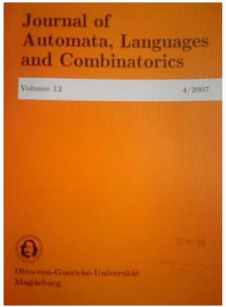
1997-2003: Team Automata and CSCW







post-2003: Team Automata and Formal Languages



Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Vector team automata

Maurice H. ter Beek [a,*], Jetty Kleijn [b]

[a] ISTI–CNR, Via G. Moruzzi 1, 56124 Pisa, Italy

[b] LIACS, Universiteit Leiden, P.O. Box 9512, 2300 RA Leiden, The Netherlands

Fundamenta Informaticae 91 (2009) 437–461

DOI 10.3233/FI-2009-0051

IOS Press

TEAMS OF PUSHDOWN AUTOMATA

MAURICE H. TER BEEK[a,*], ERZSÉBET CSUHAJ-VARJÚ[b,†] and VICTOR MITRANA[c,‡]

[a] *Istituto di Scienza e Tecnologie dell'Informazione, Consiglio Nazionale delle Ricerche, Via G. Moruzzi 1, 56124 Pisa, Italy;*

[b] *Computer and Automation Research Institute, Hungarian Academy of Sciences, Kende utca 13-17, 1111 Budapest, Hungary;*

[c] *Faculty of Mathematics, University of Bucharest, Str. Academiei 14, 70109 Bucharest, Romania*

Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 195 (2008) 41–55

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Associativity of Infinite Synchronized Shuffles and Team Automata

Maurice H. ter Beek*

Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", CNR
Via G. Moruzzi 1, 56124 Pisa, Italy
maurice.terbeek@isti.cnr.it

Jetty Kleijn†

Leiden Institute of Advanced Computer Science, Universiteit Leiden
P.O. Box 9512, 2300 RA Leiden, The Netherlands
kleijn@liacs.nl

A Calculus for Team Automata*

Maurice H. ter Beek[1]

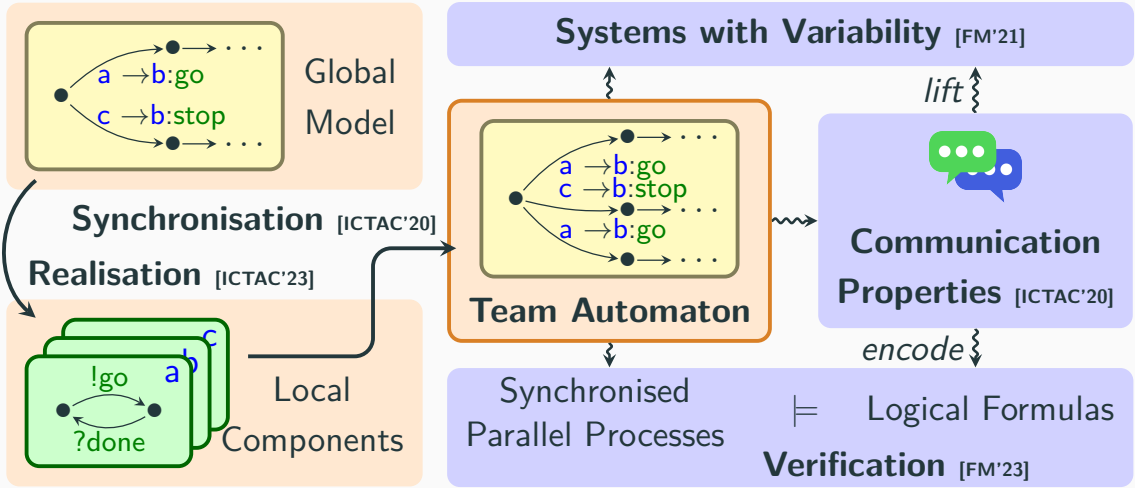
Istituto di Scienza e Tecnologie dell'Informazione, CNR via G. Moruzzi 1, 56124 Pisa, Italy

Fabio Gadducci[2]

Dipartimento di Informatica, Università di Pisa via Buonarroti 2, 56125 Pisa, Italy

Dirk Janssens[3]

post-2003: Team Automata and Formal Methods



[COORDINATION'24]

25+ Years: Selected Publications

2024	Team Automata: Overview and Roadmap	COORDINATION'24
2023	Realisability of Global Models of Interaction	ICTAC'23
2023	Overview on Constrained Multiparty Synchronisation in Team Automata *	FACS'23
2023	Can we Communicate? Using Dynamic Logic to Verify Team Automata	FM'23
2021	Featured Team Automata	FM'21
2020	Compositionality of Safe Communication in Systems of Team Automata	ICTAC'20
2020	Team Automata@Work: On Safe Communication	COORDINATION'20
2017	Communication Requirements for Team Automata	COORDINATION'17
2016	Conditions for Compatibility of Components: The Case of Masters and Slaves	ISoLA'16
2014	On Distributed Cooperation and Synchronised Collaboration	JALC
2013	Compatibility in a multi-component environment *	TCS
2012	Vector Team Automata	TCS
2010	Team Automata Based Framework for Spatio-Temporal RBAC Model *	BAIP'10
2009	Associativity of Infinite Synchronized Shuffles and Team Automata	Fundam. Inform.
2008	Extending Team Automata to Evaluate Software Architectural Design *	COMPSAC'08
2008	A calculus for team automata	ENTCS
2007	A Review on Specifying Software Architectures Using Extended Automata-Based Models *	FSEN'07
2006	Modelling a Secure Agent with Team Automata *	ENTCS
2006	A Team Automaton Scenario for the Analysis of Security Properties in Communication Protocols	JALC
2005	Team Automata for Security – A Survey –	ENTCS
2005	Modularity for Teams of I/O Automata	IPL
2004	Teams of Pushdown Automata	IJCM
2004	Interactive Behaviour of Multi-Component Systems *	ToBaCo'04
2003	Team Automata: A Formal Approach to the Modeling of Collaboration Between System Components	PhD thesis
2003	Team Automata Satisfying Compositionality	FME'03
2003	Team Automata for CSCW – A Survey – *	LNCS
2003	Synchronizations in Team Automata for Groupware Systems	CSCW
2002	Towards Team-Automata-Driven Object-Oriented Collaborative Work *	LNCS
2001	Team Automata for Spatial Access Control	ECSCW'01
1997	Team Automata for Groupware Systems	GROUP'97

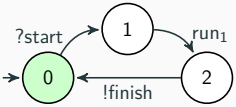
2024	Team Automata: Overview and Roadmap	COORDINATION'24
2023	Realisability of Global Models of Interaction	ICTAC'23
2023	Overview on Constrained Multiparty Synchronisation in Team Automata *	FACS'23
2023	Can we Communicate? Using Dynamic Logic to Verify Team Automata	FM'23
2021	Featured Team Automata	FM'21
2020	Compositionality of Safe Communication in Systems of Team Automata	ICTAC'20
2020	Team Automata@Work: On Safe Communication	COORDINATION'20
2017	Communication Requirements for Team Automata	COORDINATION'17
2016	Conditions for Compatibility of Components: The Case of Masters and Slaves	ISoLA'16
2014	On Distributed Cooperation and Synchronised Collaboration	JALC
2013	Compatibility in a multi-component environment *	TCS
2012	Vector Team Automata	TCS
2010	Team Automata Based Framework for Spatio-Temporal RBAC Model *	BAIP'10
2009	Associativity of Infinite Synchronized Shuffles and Team Automata	Fundam. Inform.
2008	Extending Team Automata to Evaluate Software Architectural Design *	COMPSAC'08
2008	A calculus for team automata	ENTCS
2007	A Review on Specifying Software Architectures Using Extended Automata-Based Models *	FSEN'07
2006	Modelling a Secure Agent with Team Automata *	ENTCS
2006	A Team Automaton Scenario for the Analysis of Security Properties in Communication Protocols	JALC
2005	Team Automata for Security – A Survey –	ENTCS
2005	Modularity for Teams of I/O Automata	IPL
2004	Teams of Pushdown Automata	IJCM
2004	Interactive Behaviour of Multi-Component Systems *	ToBaCo'04
2003	Team Automata: A Formal Approach to the Modeling of Collaboration Between System Components	PhD thesis
2003	Team Automata Satisfying Compositionality	FME'03
2003	Team Automata for CSCW – A Survey – *	LNCS
2003	Synchronizations in Team Automata for Groupware Systems	CSCW
2002	Towards Team-Automata-Driven Object-Oriented Collaborative Work *	LNCS
2001	Team Automata for Spatial Access Control	ECSCW'01
1997	Team Automata for Groupware Systems *	GROUP'97

2024	Team Automata: Overview and Roadmap	COORDINATION'24
2023	Realisability of Global Models of Interaction	ICTAC'23
2023	Overview on Constrained Multiparty Synchronisation in Team Automata	FACS'23
2023	Can we Communicate? Using Dynamic Logic to Verify Team Automata	FM'23
2021	Featured Team Automata	FM'21
2020	Compositionality of Safe Communication in Systems of Team Automata	ICTAC'20
2020	Team Automata@Work: On Safe Communication	COORDINATION'20
2017	Communication Requirements for Team Automata	COORDINATION'17
2016	Conditions for Compatibility of Components: The Case of Masters and Slaves	ISoLA'16
2014	On Distributed Cooperation and Synchronised Collaboration	JALC
2013	Compatibility in a multi-component environment *	TCS
2012	Vector Team Automata	TCS
2010	Team Automata Based Framework for Spatio-Temporal RBAC Model *	BAIP'10
2009	Associativity of Infinite Synchronized Shuffles and Team Automata	Fundam. Inform.
2008	Extending Team Automata to Evaluate Software Architectural Design *	COMPSAC'08
2008	A calculus for team automata	ENTCS
2007	A Review on Specifying Software Architectures Using Extended Automata-Based Models *	FSEN'07
2006	Modelling a Secure Agent with Team Automata *	ENTCS
2006	A Team Automaton Scenario for the Analysis of Security Properties in Communication Protocols	JALC
2005	Team Automata for Security – A Survey –	ENTCS
2005	Modularity for Teams of I/O Automata	IPL
2004	Teams of Pushdown Automata	IJCM
2004	Interactive Behaviour of Multi-Component Systems *	ToBaCo'04
2003	Team Automata: A Formal Approach to the Modeling of Collaboration Between System Components	PhD thesis
2003	Team Automata Satisfying Compositionality	FME'03
2003	Team Automata for CSCW – A Survey – *	LNCS
2003	Synchronizations in Team Automata for Groupware Systems	CSCW
2002	Towards Team-Automata-Driven Object-Oriented Collaborative Work *	LNCS
2001	Team Automata for Spatial Access Control	ECSCW'01
1997	Team Automata for Groupware Systems *	GROUP'97

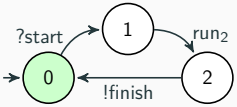
Constrained Multi-party Synchronisations

Team Automata by Example

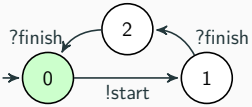
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

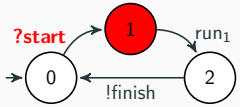
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

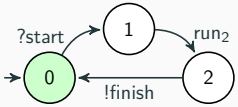
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

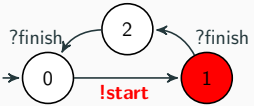
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

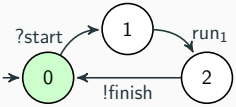
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

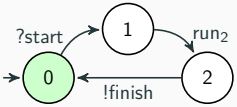
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

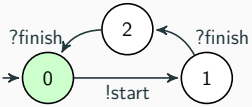
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

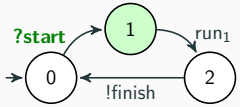
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

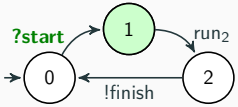
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

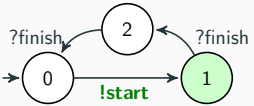
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

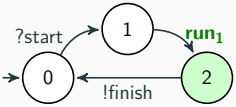
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

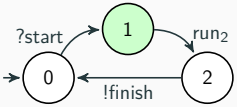
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

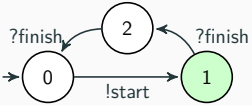
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

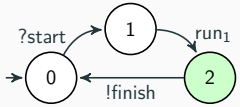
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

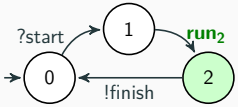
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

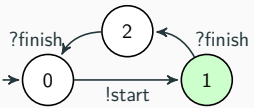
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

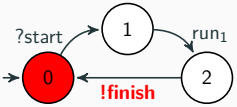
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

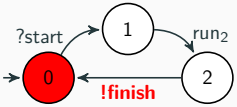
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

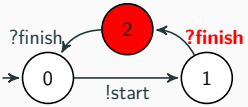
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

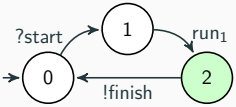
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

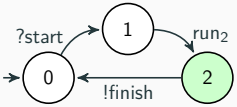
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

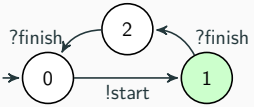
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

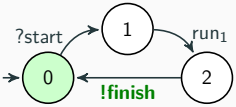
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

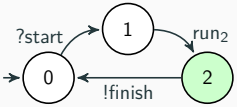
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

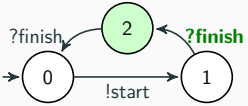
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

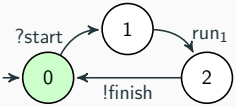
Team Automata

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

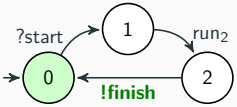
[COORDINATION'17,'20,'24] [ICTAC'20,'23]

Team Automata by Example

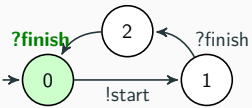
Team Automata: **not all system transitions are meaningful!**



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



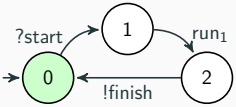
Controller

Team Automata

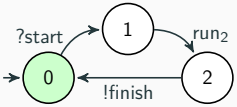
[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

[COORDINATION'17,'20,'24] [ICTAC'20,'23]

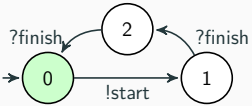
Extended Team Automata: Constrained Multi-party Synchronisations



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



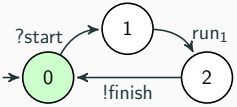
Controller

Extended TA synchronisations

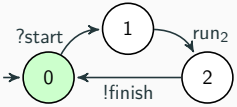
[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

[COORDINATION'17,'20,'24] [ICTAC'20,'23]

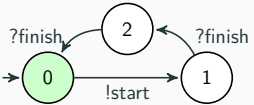
Extended Team Automata: Constrained Multi-party Synchronisations



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

Extended TA synchronisations

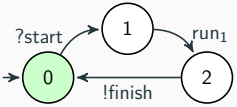
[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

[COORDINATION'17,'20,'24] [ICTAC'20,'23]

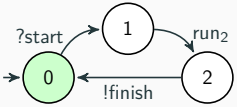
multi-party

Ctr $\rightarrow \{R1, R2\}$: start

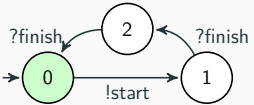
Extended Team Automata: Constrained Multi-party Synchronisations



$\mathcal{R}unner_1$



$\mathcal{R}unner_2$



Controller

Extended TA synchronisations

[CSCW'01'03] [FM'03,'21,'23] [TCS'12'13]

[COORDINATION'17,'20,'24] [ICTAC'20,'23]

multi-party

Ctr $\rightarrow \{R1, R2\}$: start

constrained

start: $1 \rightarrow 2$

finish: $1 \rightarrow 1$

Team Automata and other Coordination Models

Overview on Constrained Multi-party Synchronisation in Team Automata

and other coordination models: [FACS'23] [COORDINATION'24]

Overview on Constrained Multi-party Synchronisation in Team Automata

and other coordination models: [FACS'23] [COORDINATION'24]

Runners with orchestrators

- Reo
- BIP

↳ S.-S.T.Q. Jongmans and F. Arbab, Overview of thirty semantic formalisms for Reo. *Scientific Annals of Computer Science* 22 (2012)

S. Bliudze and J. Sifakis, The algebra of connectors: structuring interaction in BIP. *IEEE Transactions on Computers* 57 (2008)

Overview on Constrained Multi-party Synchronisation in Team Automata

and other coordination models: [FACS'23] [COORDINATION'24]

Runners with orchestrators

- Reo
- BIP

↳ S.-S.T.Q. Jongmans and F. Arbab, Overview of thirty semantic formalisms for Reo. *Scientific Annals of Computer Science* 22 (2012)

S. Bliudze and J. Sifakis, The algebra of connectors: structuring interaction in BIP. *IEEE Transactions on Computers* 57 (2008)

Runners with choreographies

- Choreography Automata
- Multi-party Session Types

↓

F. Barbanera, I. Lanese, and E. Tuosto, Choreography Automata @ COORDINATION'20

↳ S. Ghilezan, S. Jakšić, J. Pantović, A. Scalas, and N. Yoshida, Precise subtyping for synchronous multiparty sessions. *JLAMP* 104 (2019)

Overview on Constrained Multi-party Synchronisation in Team Automata

and other coordination models: [FACS'23] [COORDINATION'24]

Runners with orchestrators
- Reo
- BIP

Runners with choreographies
- Choreography Automata
- Multi-party Session Types

. . . with both
- Contract Automata

↳ S.-S.T.Q. Jongmans and F. Arbab, Overview of thirty semantic formalisms for Reo. *Scientific Annals of Computer Science* 22 (2012)

S. Bliudze and J. Sifakis, The algebra of connectors: structuring interaction in BIP. *IEEE Transactions on Computers* 57 (2008)

↓ D. Basile, P. Degano, G. Ferrari, and E. Tuosto, Relating two automata-based models of orchestration and choreography. *JLAMP* 85 (2016)

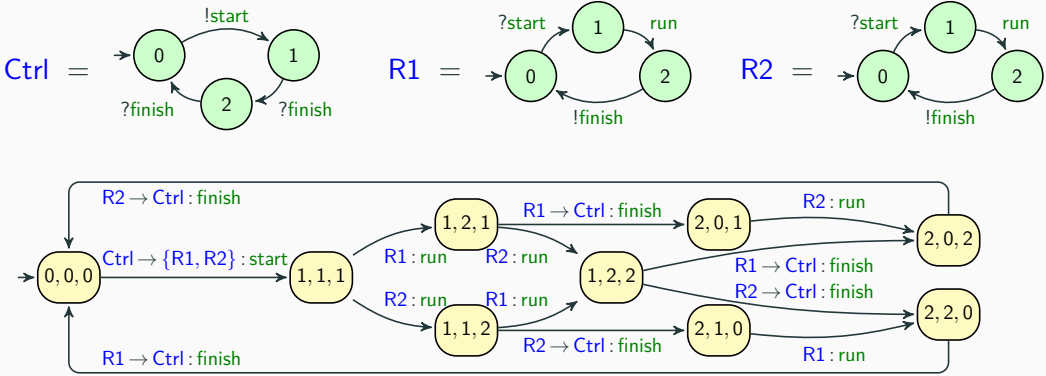
F. Barbanera, I. Lanese, and E. Tuosto, Choreography Automata @ COORDINATION'20

↳ S. Ghilezan, S. Jakšić, J. Pantović, A. Scalas, and N. Yoshida, Precise subtyping for synchronous multiparty sessions. *JLAMP* 104 (2019)

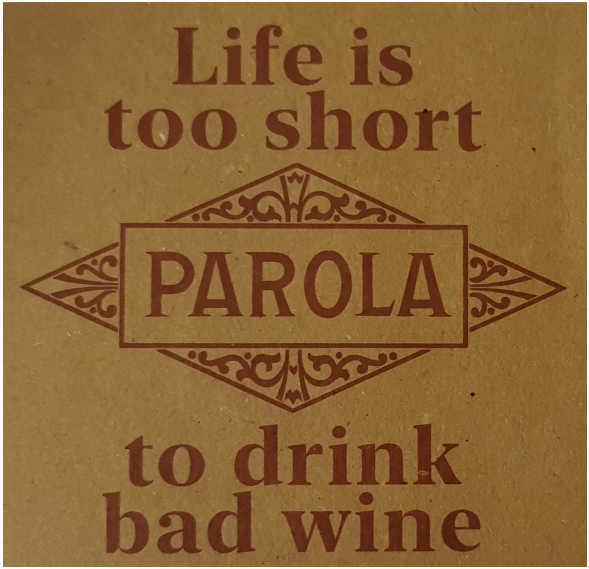
↳

In [COORDINATION'24] we discuss for each formalism (1) the definition, (2) means of composition (via synchronisation), (3) a model of the Race example, (4) a brief relation with team automata, and (5) tool support

In [COORDINATION'24] we discuss for each formalism (1) the definition, (2) means of composition (via synchronisation), (3) a model of the Race example, (4) a brief relation with team automata, and (5) tool support



Recent Results

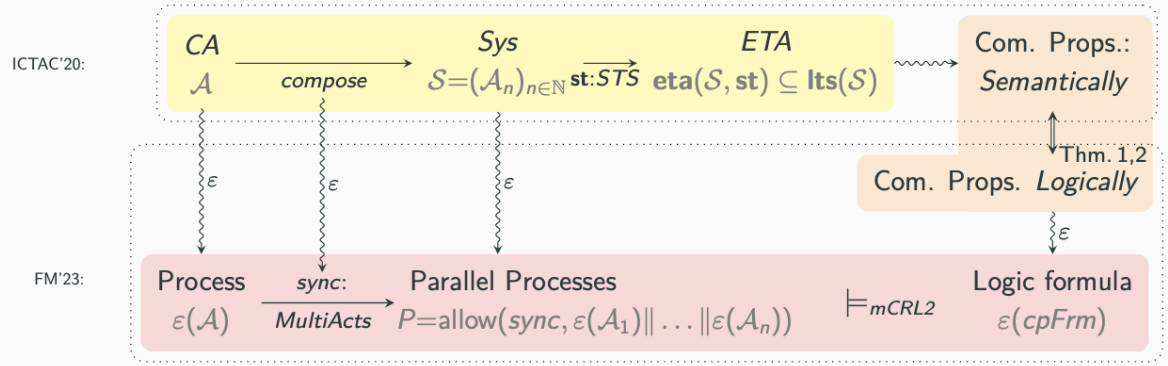

Life is
too short
PAROLA
to drink
bad wine

Model Checking Communication Properties of Team Automata

Safe Communication

Com. Props.: receptiveness (*no message loss*) & responsiveness (*no indefinite waiting*)

Com. Props.: receptiveness (*no message loss*) & responsiveness (*no indefinite waiting*)



STS *per action* in ETA: $st(a) = [\min_{out}, \max_{out}] \rightarrow [\min_{in}, \max_{in}]$

(e.g., peer-to-peer, multicast, broadcast, master-slave, ...)

mCRL2 analysing system behaviour

Online Prototype: `http://arcatools.org/feta`

ETA Specification

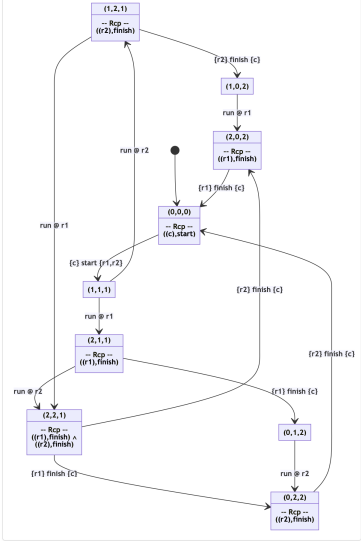
```
1  //Race example
2  CA runner (start)
3           (finish) = {
4    start 0
5    0 --> 1 by start
6    1 --> 2 by run
7    2 --> 0 by finish
8  }
9  CA controller (finish)
10          (start) = {
11   start 0
12   0 --> 1 by start
13   1 --> 2 by finish
14   2 --> 0 by finish
15 }
16 S = (r1:runner, r2:runner,
17      c:controller)
18 STS = {
19  default = 1 to 1
20  start = 1 to 2
21 }
```

Race example

ETA Examples

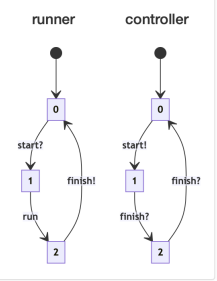
Simple | Race | Chat

ETA diagram



Demo

CA



Online Prototype: `http://arcatools.org/feta`

Communication Properties' Characterisation in mCRL2

Receptiveness:

```
[ (r1_finish|c_finish + r2_run + c_start|r1_start|r2_start + r2_finish|c_finish + r1_run)* ](
  ((<c_start> true)  =>  (<c_start|r1_start|r2_start> true)) &&
  ((<r1_finish> true)  =>  (<r1_finish|c_finish> true)) &&
  ((<r2_finish> true)  =>  (<r2_finish|c_finish> true))
)
```

Responsiveness:

```
[ (r1_finish|c_finish + r2_run + c_start|r1_start|r2_start + r2_finish|c_finish + r1_run)* ](
   (<c_finish +
     r1_start|r2_start> true)
   =>
   (<r1_finish|c_finish +
     c_start|r1_start|r2_start +
     r2_finish|c_finish> true)
)
```

Weak Receptiveness:

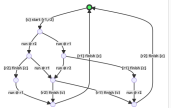
```
[ (r1_finish|c_finish + r2_run + c_start|r1_start|r2_start + r2_finish|c_finish + r1_run)* ](
  ((<r1_finish> true)  =>  (<(r2_run+r2_finish|c_finish)* . r1_finish|c_finish> true)) &&
  ((<r2_finish> true)  =>  (<(r1_finish|c_finish+r1_run)* . r2_finish|c_finish> true)) &&
  ((<c_start> true)  =>  (<(r2_run+r1_run)* . c_start|r1_start|r2_start> true))
)
```

Weak Responsiveness:

```
[ (r1_finish|c_finish + r2_run + c_start|r1_start|r2_start + r2_finish|c_finish + r1_run)* ](
   (<c_finish +
     r1_start|r2_start> true)
   =>
   (<(r2_run+r1_run)* . r1_finish|c_finish +
     c_start|r1_start|r2_start +
     (r2_run+r1_run)* . r2_finish|c_finish> true)
)
```

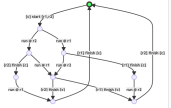
View mCRL2 evidence

Receptiveness: true

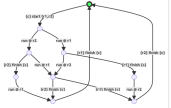


Responsiveness: false

Weak Receptiveness: true



Weak Responsiveness: true

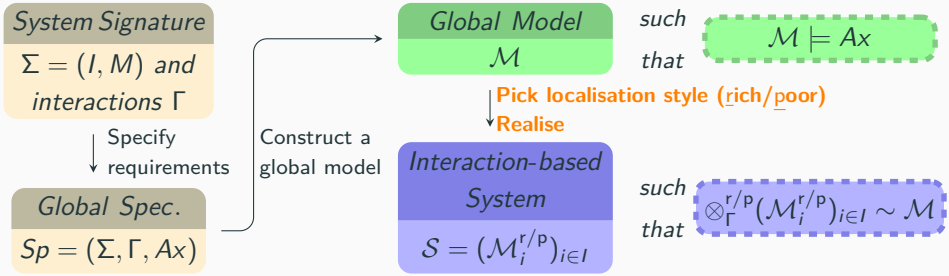


Team Automata as Realisations of Global Interaction Models

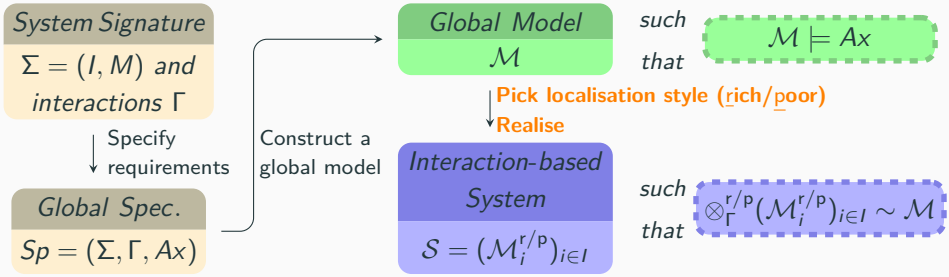
Challenge

How to check if a global model is realisable and, if it is, how to synthesise a realisation?

How to check if a global model is realisable and, if it is, how to synthesise a realisation?



How to check if a global model is realisable and, if it is, how to synthesise a realisation?



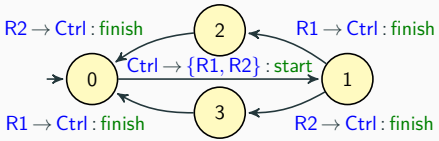
Multi-interactions

rich (à la multi-party session types, choreography languages) $i \to j : m \Rightarrow$ local output action $ij!m$ for i and local input action $ij?m$ for j

poor (à la component-based I/O development, loose coupling) $i \to j : m \Rightarrow$ local output action $!m$ for i and local input action $?m$ for j

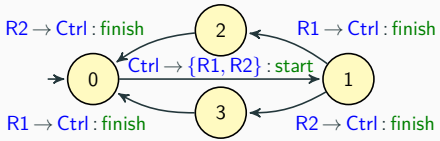
Example

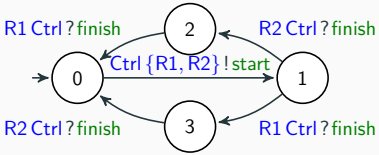
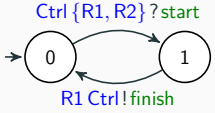
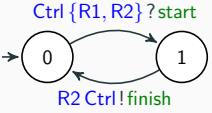
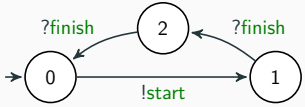
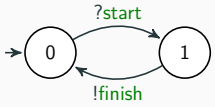
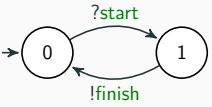
$$\Gamma_{\mathsf{Race}} = \left\{ \begin{array}{l} \mathsf{Ctrl} \to \{\mathsf{R1}, \mathsf{R2}\} : \mathsf{start}\,, \\ \mathsf{R1} \to \mathsf{Ctrl} : \mathsf{finish}\,, \\ \mathsf{R2} \to \mathsf{Ctrl} : \mathsf{finish} \end{array} \right\}$$



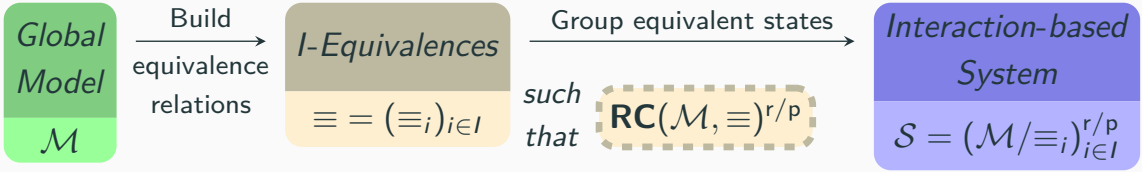
Example

$$\Gamma_{\text{Race}} = \left\{ \begin{array}{l} \text{Ctrl} \to \{\text{R1}, \text{R2}\} : \text{start}\,, \\ \text{R1} \to \text{Ctrl} : \text{finish}\,, \\ \text{R2} \to \text{Ctrl} : \text{finish} \end{array} \right\}$$



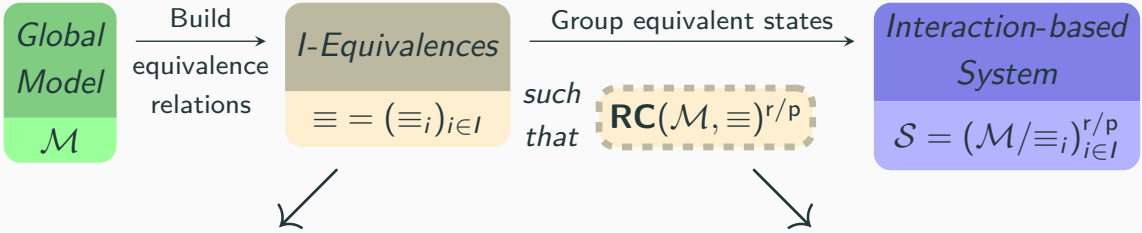
Localisation	Local Ctrl	Local R1	Local R2
Rich			
Poor			

Check Realisability and System Synthesis



Demo: `https://lmf.di.uminho.pt/ceta/`

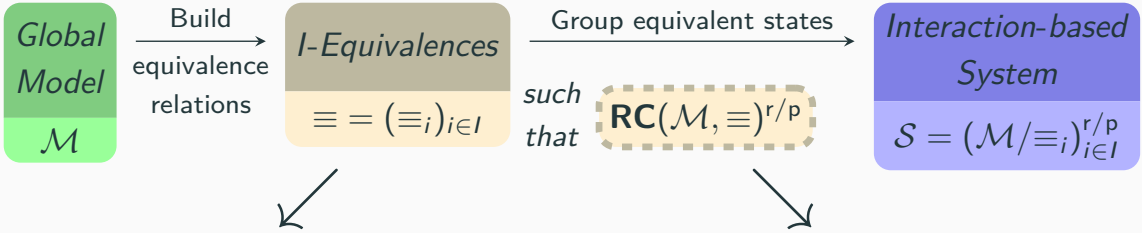
Check Realisability and System Synthesis



I. Castellani, M. Mukund, and P.S. Thiagarajan, Synthesizing Distributed Transition Systems from Global Specifications @ FSTTCS'99

cf. our paper for details:
M.H. ter Beek, R. Hennicker, and J. Proença, Realisability of Global Models of Interaction @ ICTAC'23

Check Realisability and System Synthesis



I. Castellani, M. Mukund, and P.S. Thiagarajan, Synthesizing Distributed Transition Systems from Global Specifications @ FSTTCS'99

cf. our paper for details: M.H. ter Beek, R. Hennicker, and J. Proença, Realisability of Global Models of Interaction @ ICTAC'23

Theorems 2/3 (sufficient realisability condition)
If $RC(\mathcal{M}, \equiv)^{r/p}$ holds, then $\mathcal{M} \sim \otimes_{\Gamma}^{r/p}((\mathcal{M}/\equiv_i)^{r/p})_{i \in I}$

Novelty

1. Realisations of global models with arbitrary multi-interactions supporting any kind of synchronous communication between multiple senders and multiple receivers
2. Correctness notion for realisation based on bisimulation rather than isomorhism, so allowing to deal with non-determinism
3. To construct realisations we consider, and analyse, two different localisation styles: rich and poor local actions
4. A prototypical tool Ceta checks the realisability conditions and, if they are satisfied, generates local quotients and hence realisations

`https://github.com/arcalab/choreo/tree/ceta`

`https://lmf.di.uminho.pt/ceta`

Online Prototype: https://lmf.di.uminho.pt/ceta

Choreographic Extended Team Automata

Choreography

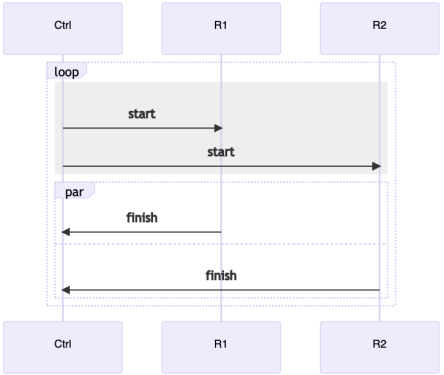
```
1 // Race example
2 (
3  (Ctrl->R1,R2: start);
4  (R1->Ctrl:finish ||
5   R2->Ctrl:finish)
6 )*
```

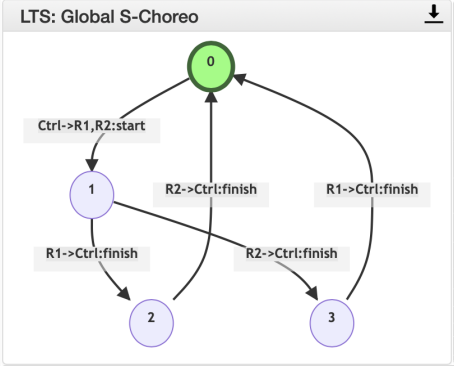
A controller starts 2 runners at the same time, and receives a finish message from each runner at a time.

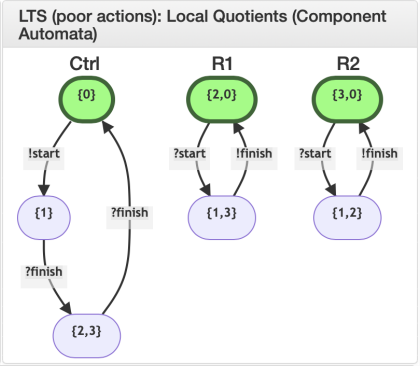
Examples

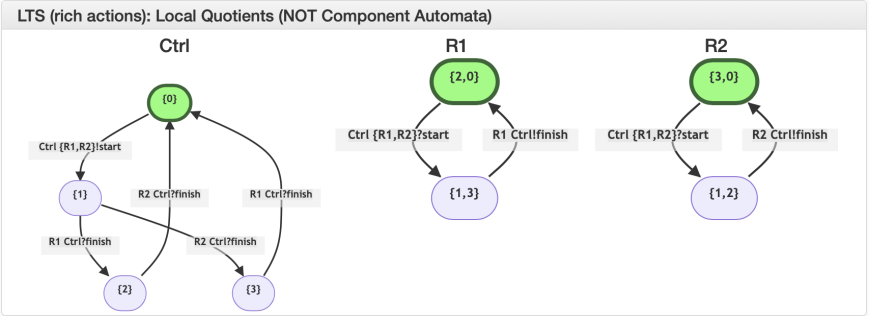
Race (simple) | Race (R1-first) | Race (once, simple) | Toss | Gossip (bad) | Gossip (good) | Cast-v1 | Cast-v2 | ab+cb+ca | ab;ac | ab|ac | ab;cd | ab|cd

Sequence Diagram




LTS: Global S-Choreo
0
Ctrl->R1,R2:start
1
R2->Ctrl:finish
R1->Ctrl:finish
R1->Ctrl:finish
R2->Ctrl:finish
2
3

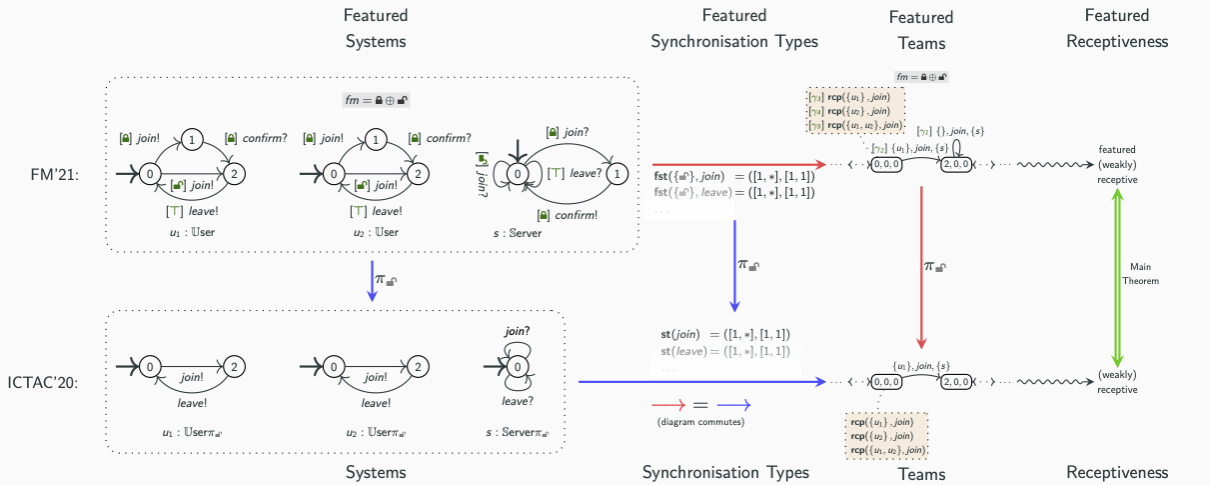

LTS (poor actions): Local Quotients (Component Automata)
Ctrl
{0}
!start
{1}
?finish
?finish
{2,3}
R1
{2,0}
?start
!finish
{1,3}
R2
{3,0}
?start
!finish
{1,2}


LTS (rich actions): Local Quotients (NOT Component Automata)
Ctrl
{0}
Ctrl {R1,R2}!start
{1}
R2 Ctrl?finish
R1 Ctrl?finish
R1 Ctrl?finish
R2 Ctrl?finish
{2}
{3}
R1
{2,0}
Ctrl {R1,R2}?start
R1 Ctrl!finish
{1,3}
R2
{3,0}
Ctrl {R1,R2}?start
R2 Ctrl!finish
{1,2}

Featured Team Automata

Featured Extended Team Automata (fETA)

Recall from safe communication: receptiveness (*no message loss*)

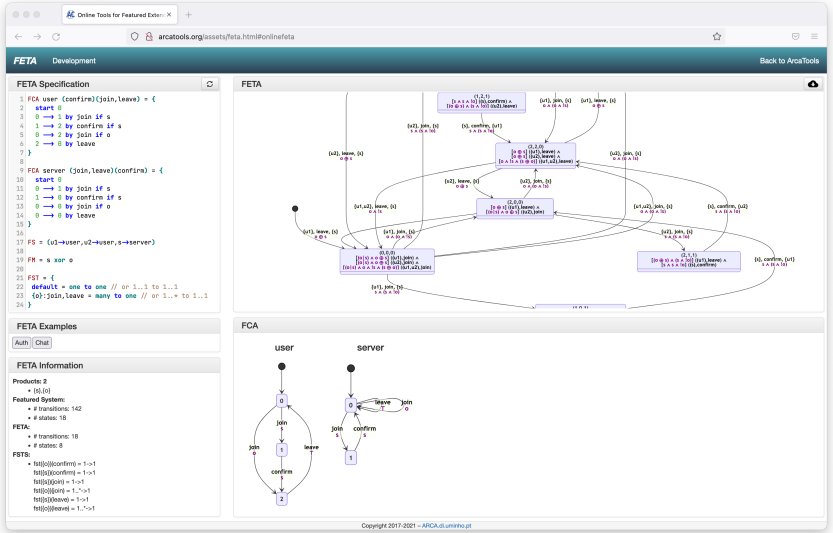


Online Prototype: `http://arcatools.org/feta`

- Specify
- Generate*
- Visualise
- Statistics

*SAT solver to solve *fm*

Demo



Ongoing Work

Team Automata with asynchronous communication?

Inspiration from multi-composition of asynchronous systems of CFSMs

D. Brand and P. Zafiropulo, On Communicating Finite-State Machines. *Journal of the ACM* 30 (1983)

F. Barbanera and R. Hennicker, Safe Composition of Systems of Communicating Finite State Machines @ ICE'24

Team Automata with asynchronous communication?

Inspiration from multi-composition of asynchronous systems of CFSMs

D. Brand and P. Zafiropulo, On Communicating Finite-State Machines. *Journal of the ACM* 30 (1983)

F. Barbanera and R. Hennicker, Safe Composition of Systems of Communicating Finite State Machines @ ICE'24

Basically finite-state I/O-automata that communicate by asynchronous exchanges of messages via buffered FIFO channels, but

- CFSM use potentially infinite FIFO buffers as message channels (like MPST)
- CFSM systems use binary peer-to-peer communication with rich local actions

Behaviour of CFSM systems formalised as a transition relation on configurations (which are pairs of a tuple of component states and a tuple of channel buffers)

Team Automata with asynchronous communication?

Inspiration from multi-composition of asynchronous systems of CFSMs

D. Brand and P. Zafiropulo, On Communicating Finite-State Machines. *Journal of the ACM* 30 (1983)

F. Barbanera and R. Hennicker, Safe Composition of Systems of Communicating Finite State Machines @ ICE'24

Basically finite-state I/O-automata that communicate by asynchronous exchanges of messages via buffered FIFO channels, but

- CFSM use potentially infinite FIFO buffers as message channels (like MPST)
- CFSM systems use binary peer-to-peer communication with rich local actions

Behaviour of CFSM systems formalised as a transition relation on configurations (which are pairs of a tuple of component states and a tuple of channel buffers)

⇒ Need to generalise to asynchronous multi-party communication for team automata

? Alternative with multisets as channels

L. Clemente, F. Herbreteau, and G. Sutre, Decidable topologies for communicating automata with FIFO and bag channels @ CONCUR'14

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

$\Rightarrow$ Need to generalise receptiveness and responsiveness to asynchronous setting

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

$\Rightarrow$ Need to generalise receptiveness and responsiveness to asynchronous setting

- Avoid orphan message configurations (in which each component is in a final state, but there is still at least one non-empty buffer)

P.-M. Deniélou and N. Yoshida, Multiparty Session Types Meet Communicating Automata @ ESOP'12

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

$\Rightarrow$ Need to generalise receptiveness and responsiveness to asynchronous setting

- Avoid orphan message configurations (in which each component is in a final state, but there is still at least one non-empty buffer) P.-M. Deniélou and N. Yoshida, Multiparty Session Types Meet Communicating Automata @ ESOP'12
- Guarantee asynchronous compatibility (in any reachable configuration, if there is a channel with a message on its top position then some component can consume it)

R. Hennicker and M. Bidoit, Compatibility Properties of Synchronously and Asynchronously Communicating Components. Logical Methods in Computer Science 14 (2018)

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

⇒ Need to generalise receptiveness and responsiveness to asynchronous setting

- Avoid orphan message configurations (in which each component is in a final state, but there is still at least one non-empty buffer) P.-M. Deniélou and N. Yoshida, Multiparty Session Types Meet Communicating Automata @ ESOP'12
- Guarantee asynchronous compatibility (in any reachable configuration, if there is a channel with a message on its top position then some component can consume it)

R. Hennicker and M. Bidoit, Compatibility Properties of Synchronously and Asynchronously Communicating Components. Logical Methods in Computer Science 14 (2018)

However, unlike team automata, they consider only binary peer-to-peer communication

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

$\Rightarrow$ Need to generalise receptiveness and responsiveness to asynchronous setting

- Avoid unspecified reception configurations (in which there is a receiving state unable to receive a message from any of its buffers, i.e., in each channel from which it could consume there is a message which it cannot receive in that state) [JACM'83]

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

$\Rightarrow$ Need to generalise receptiveness and responsiveness to asynchronous setting

- Avoid unspecified reception configurations (in which there is a receiving state unable to receive a message from any of its buffers, i.e., in each channel from which it could consume there is a message which it cannot receive in that state) [JACM'83]
- Guarantee lock-freedom (for any reachable configuration c, there's no component in a receiving state never receiving a message in all possible transition sequences from c)

F. Barbanera and R. Hennicker, Safe Composition of Systems of Communicating Finite State Machines @ ICE'24

Safety Properties of Asynchronous Team Automata?

Consider safe communication as well as safe composition, "guaranteeing the composition not to 'break' any relevant property of the single systems"

F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and E. Tuosto, Composition and decomposition of multiparty sessions. *JLAMP* 119 (2021)

$\Rightarrow$ Need to generalise receptiveness and responsiveness to asynchronous setting

- Avoid unspecified reception configurations (in which there is a receiving state unable to receive a message from any of its buffers, i.e., in each channel from which it could consume there is a message which it cannot receive in that state) [JACM'83]
- Guarantee lock-freedom (for any reachable configuration c, there's no component in a receiving state never receiving a message in all possible transition sequences from c)

F. Barbanera and R. Hennicker, Safe Composition of Systems of Communicating Finite State Machines @ ICE'24

However, unlike team automata, asynchronous communication between CFSMs uses rich local actions

How to implement Asynchronous Communication?

? Enrich synchronisation types with channel behaviour

$[1,1] \to [1,2]$: $m_1 \Rightarrow$ sync
$[2,2] \to [1,1]$: $m_2 \Rightarrow$ fifo @ receiver
$[1,1] \to [3,3]$: $m_3 \Rightarrow$ unsorted @ sender-receiver

$\Rightarrow$ Asynchronous communication with non-blocking senders

m_1 : synchronous communication between 1 sender and 1 or 2 receivers
m_2 : asynchronous communication between 2 senders and 1 receiver, and the messages are stored in a single fifo queue of the receiver
m_3 : asynchronous communication between 1 sender and 3 receivers, and the messages are stored in a multiset of both the sender(s) and the receiver

? Realisability of Asynchronous Team Automata

$\Rightarrow$ Tradeoff: more knowledge and control (sync + rich types) *versus*
more freedom and simpler systems (async + poor types)

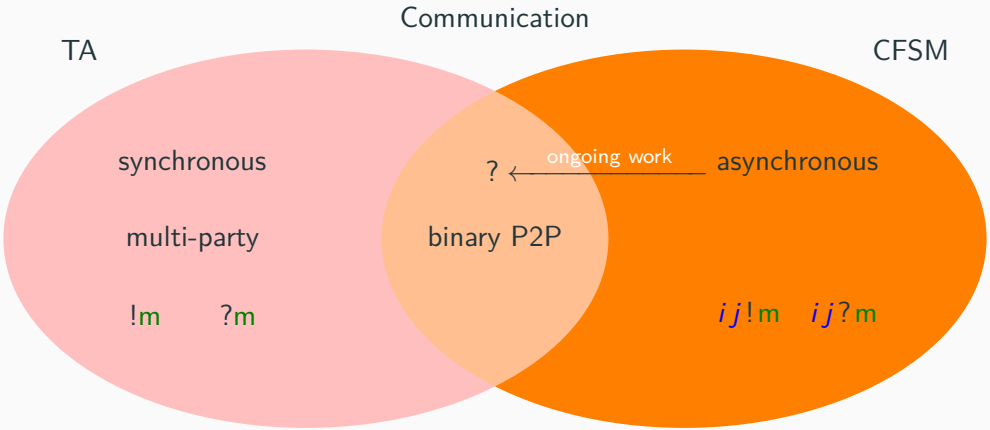
Demo:

- **Local to Global (sync+poor):**
  `http://arcatools.org/feta`
- **Global to Local (sync+rich/poor):**
  `https://lmf.di.uminho.pt/ceta/`

- **Global to Pomsets (async+rich, not TA):**
  `https://lmf.di.uminho.pt/b-pomset/`
- **Global to Scala (async+rich, not TA):**
  `https://lmf.di.uminho.pt/pompset/`
  +
  `https://lmf.di.uminho.pt/st4mp/`

Conclusion and Publicity

Conclusion



"Every good talk should include a Venn diagram" – Einar (Lima, Peru, 2023)

Thanks for your Attention!

It would be great to meet some of you at FM 2026:

- Submission: December 2nd, 2025
- Conference: May 18th–22nd, 2026
- New at FM: track on Tests and Proofs (TAP)
- Co-located: 18th International Conference on Rigorous State Based Methods (ABZ)

https://conf.researchr.org/home/fm-2026

Backup: Realisability Condition $\mathbf{RC}(\mathcal{M}, \equiv)$ sufficient but not necessary

Global model $\mathcal{M}$ does not satisfy $\mathbf{RC}(\mathcal{M}, \equiv)$, but $\mathcal{S} = \{\mathcal{M}_p, \mathcal{M}_q, \mathcal{M}_r\}$ does realise $\mathcal{M}$:

global $\mathcal{M}$	local $\mathcal{M}_p$	local $\mathcal{M}_q$	Local $\mathcal{M}_r$	$\mathbf{ta}(\mathcal{S}, \mathbf{st})$